ΤΕΧΝΟΛΟΓΙΚΟ
ΕΚΠΑΙΔΕΥΤΙΚΟ
ΙΔΡΥΜΑ
ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ

ΣΧΟΛΗ ΔΙΟΙΚΗΣΗ ΚΑΙ ΟΙΚΟΝΟΜΙΑΣ

ΠΡΩΗΝ ΤΜΗΜΑ ΕΠΙΧΕΙΡΗΜΑΤΙΚΟΥ ΣΧΕΔΙΑΣΜΟΥ ΚΑΙ ΠΛΗΡΟΦΟΡΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΤΜΗΜΑ ΔΙΟΙΚΗΣΗ ΕΠΙΧΕΙΡΗΣΕΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΤΙΤΛΟΣ: «ΣΧΕΔΙΑΣΗ ΕΦΑΡΜΟΓΗΣ ΓΙΑ ΚΙΝΗΤΕΣ ΣΥΣΚΕΥΕΣ ΠΟΥ ΝΑ ΣΤΕΛΝΕΙ ΜΕΣΩ ΔΙΑΔΙΚΤΥΟΥ ΜΗΝΥΜΑ ΣΕ ΣΥΣΤΗΜΑ ΠΟΥ ΒΑΣΙΖΕΤΑΙ ΣΕ ARDUINO ΚΑΙ ΝΑ ΜΠΟΡΕΙ ΝΑ ΕΛΕΓΧΕΙ ΤΕΣΣΕΡΙΣ ΗΛΕΚΤΡΙΚΕΣ ΣΥΣΚΕΥΕΣ»

ΣΠΟΥΔΑΣΤΗΣ: ΙΩΑΝΝΟΥ ΙΩΑΝΝΗΣ

ΕΠΟΠΤΕΥΟΝ ΚΑΘΗΓΗΤΗΣ: ΝΤΕΜΙΡΗΣ ΚΩΝΣΤΑΝΤΙΝΟΣ

ΠΑΤΡΑ-2016

ΠΡΟΛΟΓΟΣ

Στις μέρες μας η τεχνολογία είναι ένα πολύ σημαντικό κομμάτι στη ζωή μας. Παρατηρώντας τον κόσμο γύρω μας είναι εύκολο να καταλάβει κανείς ότι αυτό συμβαίνει σε μεγάλο βαθμό. Θεωρώ ότι η τεχνολογία είναι μια επιστήμη η οποία είναι και θα είναι εξελίξιμη για πολλά χρόνια ακόμα. Αναζητώντας μια είσοδο στο κόσμο της τεχνολογίας οδηγήθηκα στην αναζήτηση ενός θέματος που θα είχε σχέση με αυτή. Η κατασκευή εφαρμογών ήταν κάτι που μου άρεσε και έχω την εντύπωση πως είναι ένα καλό ξεκίνημα. Επίσης, ο μικροεπεξεργαστής του Arduino για τον οποίο όταν διάβασα πως λειτουργεί, φαντάστηκα σε πόσες κατασκευές θα μπορούσε να βρει εφαρμογή. Αυτή η φαντασία που πάντα απέχει από τη πράξη μου φάνηκε πολύ ενδιαφέρουσα και ήταν το πιο σημαντικό κίνητρο. Θα ήθελα να ευχαριστήσω την οικογένεια μου για τη σημαντική τους βοήθεια.

ΠΕΡΙΛΗΨΗ

Το θέμα της πτυχιακής μου « ΣΧΕΔΙΑΣΗ ΕΦΑΡΜΟΓΗΣ ΓΙΑ ΚΙΝΗΤΕΣ ΣΥΣΚΕΥΕΣ ΠΟΥ ΝΑ ΣΤΕΛΝΕΙ ΜΕΣΩ ΔΙΑΔΙΚΤΥΟΥ ΜΗΝΥΜΑ ΣΕ ΣΥΣΤΗΜΑ ΠΟΥ ΒΑΣΙΖΕΤΑΙ ΣΕ ARDUINO ΚΑΙ ΝΑ ΜΠΟΡΕΙ ΝΑ ΕΛΕΓΧΕΙ ΤΕΣΣΕΡΙΣ ΗΛΕΚΤΡΙΚΕΣ ΣΥΣΚΕΥΕΣ » είναι ένα τίποτα μπροστά σε όλα αυτά που σχεδιάζονται υλοποιούνται καθημερινά γύρω μας («έξυπνα σπίτια», διάφορες εφαρμογές, γρήγορες επικοινωνίες, ιδέες που βγαίνουν καθημερινά. Η έρευνα του θέματος είναι μια είσοδος σε όλα αυτά καθώς μας «αναγκάζει» να εισχωρήσουμε σε γνώσεις λειτουργικού-υλικού, πως επικοινωνούν μεταξύ τους, σχεδίασης μεθοδολογίας και τρόπους σύνδεσης. Πιο συγκεκριμένα το υλικό αποτελείται από τη πλατφόρμα του ARDUINO μαζί με τις shield που μπορεί να χρησιμοποιήσει (Ethernet shield), ηλεκτρονόμους (relay), τρανζίστορ, καλώδια, ηλεκτρικές συσκευές. Το λειτουργικό αποτελείται από τα εργαλεία που θα μελετήσουμε και θα χρειαστούμε (ANDROID STUDIO, IDE). Η σχεδίαση και η μεθοδολογία έχει να κάνει με τον τρόπο που θα επικοινωνούν θα συνδεθούν όλα αυτά μεταξύ τους αλλά και το επιθυμητό αποτέλεσμα που είναι η επίτευξη του έργου που περιγράφει το θέμα. Στο τέλος θα παρουσιαστεί το τελικό αποτέλεσμα μαζί με τις παρατηρήσεις του. Πιο σημαντική παρατήρηση θεωρώ είναι το πώς θα μπορούσε να εξελιχτεί το τελικό αποτέλεσμα.

SUMMARY

The subject of my dissertation "DESIGN APPLICATION FOR MOBILE DEVICES TO SEND ONLINE MESSAGE IN A SYSTEM BASED ON ARDUINO as a controller of FOUR APPLIANCES" is nothing compared to all that are designed every day around us ("smart homes" , various applications, rapid communications, ideas that come out daily). The collection of information for the subject help us to “enter” in operating-material knowledge, how becomes the cumminication between them, design methodology and ways of connecting. More specifically, the material consists of the ARDUINO platform together with the shield that can be used (Ethernet shield), relays (relay), transistors, wires, electrical appliances. The function consists of the tools that we will study and will need (ANDROID STUDIO, IDE). The design and methodology has to do with the way you communicate will connect all this together and the desired result is the achievement of the project describing the subject. At the end we will present the final result together with his comments. Most important observation is how I could evolve the final result.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΕΦΑΛΑΙΟ 1^{ο} – ΕΙΣΑΓΩΓΗ

Υπάρχουν πολλές χρήσιμες εφαρμογές για κινητές συσκευές οι οποίες εξυπηρετούν διάφορες ανάγκες. «Σήμερα» η ανάγκη του ανθρώπου για καινοτόμα επιχειρηματικά σχέδια, τεχνολογίες, εξέλιξη της οικονομίας επιβάλει και οδηγεί σε τέτοιες αναζητήσεις για δημιουργία εφαρμογών που θα βελτιώσει τον τρόπο ζωής αυτών που τις χρησιμοποιούν.

Η ανάπτυξη μιας εφαρμογής προϋποθέτει γνώσεις σχεδίασης και προγραμματισμού. Να σημειώσουμε ότι βασική προϋπόθεση για να ξεκινήσει μια τέτοια διαδικασία είναι να γνωρίζουμε τι ακριβώς είναι αυτό που θέλουμε να εξυπηρετεί και με ποιο τρόπο. Στη δικιά μας περίπτωση, το πρόβλημα που θέλουμε να λύσουμε ή αλλιώς ανάγκη που θέλουμε να ικανοποιήσουμε είναι το πώς ένας άνθρωπος θα ελέγχει 4 ηλεκτρικές συσκευές της οικίας του από οποιοδήποτε σημείο του κόσμου. Ο τρόπος που θα αναζητήσουμε για να λύσουμε αυτό το πρόβλημα είναι η υλοποίηση μιας εφαρμογής για κινητά η οποία μέσω του διαδικτύου θα επικοινωνεί με τις συσκευές στην οικία.

Πριν χρόνια κάτι τέτοιο, θα θεωρούταν αδύνατο αλλά η τεχνολογία έχει κάνει τεράστια βήματα και στις μέρες μας το συγκεκριμένο θέμα αποτελεί γεγονός. Άνθρωποι που έχουν ικανοποιήσει τις βασικές τους ανάγκες και έχουν την οικονομική δυνατότητα να ικανοποιήσουν και δεύτερες, στρέφουν το ενδιαφέρον τους στο πως θα κάνουν τη ζωή τους πιο άνετη. Στις μέρες μας, δεν είναι λίγοι αυτοί που έχουν τη δυνατότητα μέσα από το κινητό τους ή το tablet τους από οποιοδήποτε σημείο, μέσω του διαδικτύου να έχουν πρόσβαση στο σπίτι τους. Οι λόγοι που υπάρχει η εξέλιξη σε αυτόν το τομέα είναι τα συναισθήματα όπως η ασφάλεια που νιώθει κάποιος που ενώ λείπει από το σπίτι του μπορεί να ενεργοποιεί ή να απενεργοποιεί το συναγερμό, να βλέπει τι γίνεται σε αυτό μέσω αισθητήρων και συστημάτων παρακολούθησης, η αίσθηση της οικονομίας καθώς μπορεί να κλείνει τα φώτα, τη τηλεόραση, το θερμοσίφωνο αν τα ξέχασε σε λειτουργία, η άνεση καθώς ενώ δεν είναι στην οικία του, μέσω της εφαρμογής που προαναφέραμε και άλλων παραμέτρων υλικού, ζεσταίνει το φαγητό του χωρίς να είναι στο σπίτι ώστε να το βρει έτοιμο όταν γυρίσει ή ανάβει τη καφετέρια για να βρει τον καφέ του έτοιμο. Συμπεραίνει κανείς εύκολα ότι υπάρχει αρκετό ενδιαφέρον να δούμε πως μπορεί να σχεδιαστεί και να υλοποιηθεί ένα τέτοιο σχέδιο.

Το θέμα μας, «ο έλεγχος τεσσάρων ηλεκτρικών συσκευών μέσω εφαρμογής», αποτελεί θα λέγαμε μικρογραφία ενός «έξυπνου» σπιτιού αλλά δεν παύει να μας βοηθάει να εισχωρήσουμε στη τεχνολογία και τεχνοτροπία για το πώς γίνεται ένα «έξυπνο σπίτι». Οι τέσσερις συσκευές που θα χρησιμοποιηθούν στο σχέδιο μας έχουν να κάνουν με την άνεση και την οικονομία και είναι οι εξής: δύο φωτιστικά γραφείου (λάμπες), ένα αερόθερμο και ένα ραδιόφωνο. Αφού αποφασίσαμε και απαντήσαμε στο ερώτημα ποιες θα είναι οι τέσσερις συσκευές, το επόμενο ερώτημα που γεννιέται είναι ποιος θα είναι ο τύπος (λειτουργικό σύστημα) του κινητού που θα χρησιμοποιήσουμε. Θα χρησιμοποιήσουμε κινητά με λειτουργικό ANDROID καθώς

υποστηρίζουν τον «ανοιχτό κώδικα» και δίνουν την ελευθερία σε οποιονδήποτε έχει τις απαραίτητες γνώσεις ή μη να επεμβαίνει στο λειτουργικό σύστημα και να κάνει ότι θέλει. Για να υλοποιηθεί ένα τέτοιο σχέδιο, θα χρειαστούμε επίσης για τον έλεγχο των συσκευών τη συσκευή(μικροεπεξεργαστή) ARDUINO η οποία κυκλοφορεί σε πολλές εκδόσεις και έχει πολλές επεκτάσεις ανάλογα με τον τρόπο που θα χρησιμοποιηθεί. Η σωστή σύνδεση και σχεδίαση όλων των κομματιών που συνθέτουν ένα τέτοιο σχέδιο θα οδηγήσουν ύστερα στην υλοποίηση του.

Πριν ξεκινήσω να δακτυλογραφώ την πτυχιακή μου, συλλέγοντας πολλές πληροφορίες για το θέμα μου και έχοντας κάποιες αποτυχίες από την σχεδίαση του μέχρι και την πράξη στη πορεία να υλοποιήσω το σχέδιο, αντιλήφθηκα ότι αυτά τα λάθη και αυτές οι αποτυχίες αν εκφραστούν σωστά και κατανοητά σε αυτό το κείμενο, θα βοηθήσουν κάποιον που θα το διαβάσει και θα θέλει να κάνει κάποιο παρόμοιο σχέδιο σε μεγάλο βαθμό ώστε να μην κάνει τα ίδια λάθη και να κερδίσει χρόνο. Θεωρώ επίσης ότι η εργασία εκτός από στοιχεία καθοδηγητικού χαρακτήρα για το συγκεκριμένο σχέδιο, θα υποκινήσει νέες ιδέες προς πολλές κατευθύνσεις και σχέδια μιας και η ανάπτυξη προγραμματισμού, εφαρμογών είναι απαραίτητη σε πολλούς τομείς της δημιουργίας (εύρεση-ανάπτυξη άλλων καινοτόμων εφαρμογών που θα κάνουν πιο εύκολη τη ζωή μας) όπως και η χρήση της συσκευής του ARDUINO η οποία με τις προσφερόμενες επεκτάσεις της μπορεί να χρησιμοποιηθεί σε οτιδήποτε, από έναν απλό έλεγχο συσκευών μέχρι ρομποτική.

Στα επόμενα κεφάλαια θα σας παρουσιάσω την συσκευή του ARDUINO μαζί με τις εκδόσεις και τις διαφορές τους αλλά και όλες τις επεκτάσεις της όπως και το πώς χρησιμοποιούνται και συνδέονται όλα αυτά. Επίσης θα σας παρουσιάσω αναλυτικά τις τεχνικές αλλαγές (παραμετροποίηση των τεσσάρων συσκευών) που πρέπει να πραγματοποιήσουμε καθώς επίσης κι όλο το υλικό που θα χρησιμοποιηθεί (τύποι καλωδίων, relays,transzistor). Ύστερα, θα σχεδιάσουμε τη συνδεσμιολογία των συσκευών με το ARDUINO και πως όλα αυτά συνδέονται στο διαδίκτυο. Στα τελευταία κεφάλαια θα εισχωρήσουμε στο κόσμο των program developer και θα παρουσιαστεί ο κώδικας προγραμματισμού μαζί με το αποτέλεσμα του (τι εμφανίζει η οθόνη του mobile) για την εφαρμογή που θα εγκατασταθεί στο ANDROID mobile phone αφού πρώτα το αποφασίσουμε σχεδιάσουμε καθώς και ο κώδικας προγραμματισμού για τη συσκευή του ARDUINO μαζί με ότι άλλο χρειάζεται να ξέρουμε για τις γλώσσες προγραμματισμού που χρησιμοποιούνται στο κάθε κώδικα και πως αυτές επικοινωνούν μεταξύ τους.

ΚΕΦΑΛΑΙΟ 2^{Ο} –ARDUINO, ETHERNET SHIELD

2.1 ARDUINO

2.1.1 ΓΝΩΡΙΖΟΝΤΑΣ ΤΟ ARDUINO

Σε αυτό το σημείο, θα ήθελα να σας περιγράψω την γνωριμία μου με απλούς όρους με τη πλατφόρμα ARDUINO και ύστερα να συνεχίσω με πιο ειδικούς. Μια πλατφόρμα σαν αυτή συνδέεται με το ρεύμα, έχει ανάλογα με την έκδοση της κάποια μνήμη, μικροελεγκτή, θύρες εισόδου και εξόδου. Μπορεί να τροφοδοτείται με ρεύμα μέσω ενός υπολογιστή ή ακόμα και μέσω μιας μπαταρίας ή κάποιας από τις επεκτάσεις της που θα περιγράψω σε επόμενη παράγραφο. Συνδέοντας τον ARDUINO με έναν υπολογιστή, εκτός από το ότι τη τροφοδοτούμε με ρεύμα, έχουμε τη δυνατότητα αφού εγκαταστήσουμε το κατάλληλο λογισμικό-πρόγραμμα στον υπολογιστή αυτό, να γράψουμε εγκαταστήσουμε κάποια προγράμματα (skertch) στη μνήμη της πλακέτας. Αυτά τα προγράμματα αφορούν τον τρόπο λειτουργίας της πλακέτας , πιο συγκεκριμένα τις θύρες εισόδου και εξόδου και πως, πότε αυτές λειτουργούν. Αυτές τις λειτουργίες, εντολές που περιγράφουν τα προγράμματα, είναι «δουλειά» του μικροελεγκτή να τις πραγματοποιήσει. Ανάλογα με το ποιες είναι οι εντολές αυτές, οι θύρες εισόδου, εξόδου μεταφέρουν δεδομένα ψηφιακά ή αναλογικά. Θέλοντας να υποκινήσω το ενδιαφέρον σας, θα περιέγραφα τη πλατφόρμα αυτή σαν ένα «μυαλό» το οποίο ανάλογα με τις δυνατότητες της κάθε έκδοσης της πλατφόρμας έχει περισσότερες ή λιγότερες δυνατότητες. Ένα «μυαλό» το οποίο αν κάποιος έχει τις κατάλληλες γνώσεις και τεχνογνωσία μπορεί να το ρυθμίσει να κάνει ότι θέλει μέσα στα λογικά όρια. Με πιο ειδικούς όρους, ο **Arduino** είναι ένας single-board μικροελεκτής, δηλαδή μια μητρική πλακέτα ανοικτού κώδικα, με ενσωματωμένο μικροελεγκτή και εισόδους/εξόδους, όπως ανέφερα, η οποία μπορεί να προγραμματιστεί με τη γλώσσα Wiring (ουσιαστικά πρόκειται για τη γλώσσα προγραμματισμού C++ και ένα σύνολο από βιβλιοθήκες, υλοποιημένες επίσης στην C++). Το Arduino μπορεί να χρησιμοποιηθεί για την ανάπτυξη ανεξάρτητων διαδραστικών αντικειμένων αλλά και να συνδεθεί με υπολογιστή μέσω προγραμμάτων σε Processing, Max/MSP, Pure Data, SuperCollider.

2.1.2 Πως δημιουργήθηκε-Ιστορικό

Το 2005, ένα σχέδιο κίνησε προκειμένου να φτιαχτεί μία συσκευή για τον έλεγχο προγραμμάτων διαδραστικών σχεδίων από μαθητές, η οποία θα ήταν πιο φθηνή από άλλα πρωτότυπα συστήματα διαθέσιμα εκείνη την περίοδο. Οι ιδρυτές Massimo Banzi και David Cueartielles ονόμασαν το σχέδιο από τον Arduin της Ivrea και ξεκίνησαν να παράγουν πλακέτες σε ένα μικρό εργοστάσιο στην Ιβρέα, κωμόπολη της επαρχίας Τορίνο στην περιοχή Πεδεμόντιο της βορειοδυτικής Ιταλίας- την ίδια περιοχή στην οποία στεγαζόταν η εταιρία υπολογιστών Olivetti.

ΕΚΔΟΣΕΙΣ-ΕΞΕΛΙΞΗ

- Τον Σεπτέμβριο του 2006 ανακοινώθηκε το Arduino Mini
- Τον Οκτώβρη του 2008 ανακοινώθηκε το Arduino Duemilanove. Αρχικά βασίστηκε στο Atmel Atmega168, αλλά μετά στάλθηκε με το ATmega328[3].
- Τον Μάρτιο του 2009 ανακοινώθηκε το Arduino Mega. Είναι βασισμένο στο Atmel ATmega1280
- Από τον Μάιο του 2011 πάνω από 300,000 Arduino ήταν σε χρήση σε όλο τον κόσμο
- Τον Ιούλιο του 2012 ανακοινώθηκε το Arduino Leonardo. Είναι βασισμένο στο Atmel ATmega32u4
- Τον Οκτώβριο του 2012 ανακοινώθηκε το Arduino Due. Είναι βασισμένο στο Atmel SAM3X8E, που είχε πυρήνα
- Τον Νοέμβριο του 2012 ανακοινώθηκε το Arduino Micro. Είναι βασισμένο στο Atmel ATmega32u4
- Τον Μάιο του 2013 ανακοινώθηκε το Arduino Robot. Είναι βασισμένο στο Atmel ATmega32u4 και ήταν το πρώτο επίσημο Arduino με ρόδες
- Τον Μάιο του 2013 ανακοινώθηκε το Arduino Yun. Είναι Βασισμένο στο ATmega32u4 και στο Atheros AR9331 και ήταν το πρώτο προϊόν wifi που συνδύαζε το Arduino με το Linux.

2.1.3 ΜΟΝΤΕΛΑ ARDUINO

Η πλατφόρμα του ARDUINO είναι απαραίτητη για την υλοποίηση του σχεδίου μας. Σημαντικότερο παράγοντα στην επιλογή του μοντέλου που θα χρησιμοποιήσουμε, είναι η σχέση αξιοποίησης δυνατοτήτων της πλατφόρμας με το

κόστος. Δηλαδή επιλέγουμε το πιο φτηνό αρκεί να μας παρέχει όλα όσα χρειαζόμαστε και προσοχή φυσικά να είναι η γνήσια έκδοση. Οι διαφοροποιήσεις τους έχουν να κάνουν με τους αναλογικούς και ψηφιακούς δέκτες, τον μικροελεγκτή, τη τάση λειτουργίας, τη μνήμη και στη ταχύτητα του ρολογιού. Οι εταιρείες που κατασκευάζουν πλατφόρμες Arduino είναι η Ιταλική εταιρία Smart Projects και η Αμερικάνικη εταιρεία SparkFun Electronics.

Τα μοντέλα με τα ονόματα και τα χαρακτηριστικά τους παραθέτονται στο πίνακα 1.

me	**Processor**	**Operating/Input Voltage**	**CPU Speed**	**Analog In/Out**	**Digital IO/PWM**	**EEPROM [kB]**	**SRAM [kB]**	**Flash [kB]**	**USB**	**UA**
l	Intel Curie ®	3.3 V/ 7-12V	32MHz	6/0	14/4	-	24	196	Regular	-
e	ATSAM3X8E	3.3 V / 7-12 V	84 MHz	12/2	54/12	-	96	512	2 Micro	4
mma	ATtiny85	3.3 V / 4-16 V	8 MHz	1/0	3/2	0.5	0.5	8	Micro	0
yPad	ATmega168V ATmega328P	2.7-5.5 V / 2.7-5.5 V	8MHz	6/0	14/6	0.512	1	16	-	-
yPad npleSnap	ATmega328P	2.7-5.5 V / 2.7-5.5 V	8 MHz	4/0	9/4	1	2	32	-	-
yPad B	ATmega32U4	3.3 V / 3.8-5 V	8 MHz	4/0	9/4	1	2.5	32	Micro	-
ga 2560	ATmega2560	5 V / 7-12 V	16 MHz	16/0	54/15	4	8	256	Regular	4
ga ADK	ATmega2560	5 V / 7-12 V	16 MHz	16/0	54/15	4	8	256	Regular	4
cro	ATmega32U4	5 V / 7-12 V	16 MHz	12/0	20/7	1	2.5	32	Micro	1
KR1000	SAMD21 Cortex-M0+	3.3 V/ 5V	48 MHz	7/1	8/4	-	32	256	Micro	1
no	ATmega168V	5 V / 7-9 V	16	8/0	14/16	0.512	1	16	-	-

	ATmega328P		MHz							
ɔ	ATmega168V ATmega328P	5 V / 7-9 V	14 16 MHz	6/0	14/16	0.512 1	1 2	16 32	-	1
ɔ Mini	ATmega328P	3.3 V / 3.35-12 V 5 V / 5-12 V	8 MHz 16 MHz	6/0	14/6	1	1	32	-	1
o	ATmega328P	5 V / 7-12 V	16 MHz	16/0	14/6	1	2	32	Regular	1
n	ATmega32U4 AR9331 Linux	16 MHz 400 MHz	12/0	20/7	1	2.5	16	32 64 MB	Micro 1	1
ro	ATSAMD21G18	3.3 V / 7-12 V	48 MHz	6/1	14/10	-	32	256	2 Micro	2

Πίνακας 1.

2.2 ΕΠΙΛΟΓΗ ΜΟΝΤΕΛΟΥ-ARDUINO UNO GENUINE

ΕΠΙΛΟΓΗ ΜΟΝΤΕΛΟΥ

Βάση του σχεδίου που θα σας παρουσιάσω παρακάτω, εκτός από τα υπόλοιπα που επίσης θα δούμε, θα χρησιμοποιήσω το μοντέλο της πλατφόρμας με την ονομασία ARDUINO UNO Genuine και επίσης τη shield με ονομασία Ethernet shield. Ο λόγος που χρησιμοποίησα αυτό το μοντέλο πλατφόρμας, είναι ότι οι δυνατότητες του είναι επαρκείς για το σχέδιο μου, η τιμή του δεν είναι ιδιαίτερα υψηλή. Επίσης είναι σημαντικό να αναφέρω ότι το site του arduino τη προτείνει σε κάποιον που ασχολείται πρώτη φορά με ένα τέτοιο σχέδιο ή κάτι παρόμοιο.

Θεωρώ σε αυτό το σημείο, ότι είναι σημαντικό να περιγράψω τη πλατφόρμα arduino UNO καθώς να αναφέρω και να αναλύσω τα χαρακτηριστικά της μιας και είναι απαραίτητο κάτι τέτοιο αφού στη συνέχεια θα επεκταθώ σε πιο ειδικά κομμάτια του σχεδίου.

ΕΙΣΑΓΩΓΗ ΣΤΗ ΠΛΑΤΦΟΡΜΑ UNO Genuine

Το UNO Genuino *(Εικόνα 1)* είναι μια πλακέτα μικροελεγκτή με βάση τον επεξεργαστή ATmega328P. Έχει 14 ψηφιακές καρφίτσες εισόδου / εξόδου (εκ των οποίων 6 μπορούν να χρησιμοποιηθούν ως έξοδοι PWM), 6 αναλογικές εισόδους, έναν 16 MHz CPU, μια σύνδεση USB, μια υποδοχή ρεύματος, μια κεφαλίδα ICSP και ένα κουμπί επαναφοράς. Περιέχει όλα όσα χρειάζονται για τη στήριξη του μικροελεγκτή.

"Uno" σημαίνει “ένα” στα ιταλικά και επιλέχτηκε για να σηματοδοτήσει την κυκλοφορία του λογισμικού Arduino(IDE) 1.0 το οποίο όπως θα δούμε είναι απαραίτητο για το σχέδιο μας. Το μοντέλο Uno και η έκδοση Arduino IDE 1.0 του Arduino λογισμικού (IDE) θεωρούνται εκδόσεις αναφοράς για τη συσκευή του Arduino και τους κατασκευαστές του και πλέον έχουν εξελιχθεί σε νεότερες εκδόσεις και ο λόγος είναι ότι το μοντέλο Uno είναι η πρώτη από μια σειρά USB πλατφόρμες Arduino.

Η USB θύρα χρησιμοποιείται για την σύνδεση της πλατφόρμας με τον υπολογιστή και τον προγραμματισμό του. Ο έλεγχος και ο καθορισμός των εσόδων εξόδων γίνεται μέσω των εντολών που γράφουμε στο πρόγραμμα όπως pinmode( ), digitalwrite( ) και digitalread( ). Σε επόμενο κεφάλαιο θα γίνει περαιτέρω ανάλυση όσον αφορά την γλώσσα προγραμματισμού και τις εντολές που χρησιμοποιούνται. Η πλατφόρμας μας λειτουργεί στα 5 V με δυνατότητα διαχείρισης 40 mA σε κάθε είσοδο/έξοδο ακροδέκτη. Κάθε ακροδέκτης διαθέτει εσωτερικό αντιστάτη pull up με τιμή αντίστασης 20-50 ΚΩ.

(Εικόνα 1. ARDUINO Genuino)

ΤΕΧΝΙΚΑ ΧΑΡΑΚΤΗΡΗΣΤΙΚΑ

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 Ma
DC Current for 3.3V Pin	50 Ma
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader

SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Για να προγραμματίσουμε τον ARDUINO ή αλλιώς να κάνουμε upload τα προγράμματα μας πρέπει να εγκαταστήσουμε στον υπολογιστή που θα συνδέσουμε με την πλατφόρμα μέσω της USB θύρας της, το λογισμικό ARDUINO IDE το οποίο υπάρχει διαθέσιμο από διάφορες πηγές στο διαδίκτυο.

Ο επεξεργαστής ATmega328 στο Uno έρχεται προγραμματισμένος με ένα bootloader που μας επιτρέπει να ανεβάσουμε νέο κώδικα σε αυτό χωρίς τη χρήση ενός εξωτερικού προγραμματιστή υλικού. Μπορούμε επίσης να παρακάμψουμε τον bootloader και τον προγραμματισμό του μικροελεγκτή και να κάνουμε άλλες ενέργειες αλλά στη περίπτωση μας όπως θα δούμε δεν χρειάζεται κάτι τέτοιο.

ΧΡΗΣΙΜΕΣ ΠΛΗΡΟΦΟΡΙΕΣ

Η Uno έχει δυνατότητα επαναφοράς της σύνδεσης η οποία προστατεύει τις θύρες USB του υπολογιστή σας από βλάβες και υπερένταση. Αν και οι περισσότεροι υπολογιστές παρέχουν τη δική τους εσωτερική προστασία, η ασφάλεια παρέχει ένα επιπλέον επίπεδο προστασίας. Εάν περισσότερα από 500 mA εφαρμοστούν στην θύρα USB, η σύνδεση θα σταματήσει μέχρι να μην υπάρχει υπερφόρτωση.

ΤΡΟΦΟΔΟΤΗΣΗ

Η πλατφόρμα Uno μπορεί να τροφοδοτείται μέσω της σύνδεσης USB ή με εξωτερικό τροφοδοτικό. Η πηγή ενέργειας επιλέγεται αυτόματα.

Εκτός από τη θύρα USB μπορεί να προέλθει από ένα AC-σε-DC προσαρμογέα ή μια μπαταρία. Ο προσαρμογέας μπορεί να συνδεθεί με τη σύνδεση ενός 2,1 χιλιοστά κέντρο θετικό βύσμα στην υποδοχή τροφοδοσίας της πλατφόρμας. Τα «+» και «-»

από μια μπαταρία μπορούν να συνδεθούν στο GND και Vin pin headers της σύνδεσης POWER.

Σημειώνεται ότι η πλατφόρμα μπορεί να λειτουργήσει με εξωτερική πηγή 6-20 βολτ. Εάν συνδεθεί με λιγότερο από 7V , το pin 5V μπορεί να παρέχει λιγότερο από πέντε βολτ και τότε ίσως η πλατφόρμα να είναι «ασταθής» όσον αφορά τη λειτουργία της. Εάν χρησιμοποιηθούν περισσότερα από 12V, ο ρυθμιστής τάσης μπορεί να υπερθερμανθεί και να προκληθεί ζημιά στη πλατφόρμα μας. Η συνιστώμενη σειρά είναι 7 έως 12 βολτ.

Οι ακροδέλτες τροφοδοσίας είναι ως εξής:

- Ακροδέκτης Vin. Η είσοδος τάσης του ρεύμαμτος στη Uno board, όταν θέλουμε να χρησιμοποιήσουμε μια εξωτερική πηγή ενέργειας (σε αντιδιαστολή με 5 V από τη σύνδεση USB ή άλλες οργανωμένες πηγή ενέργειας).
- Ακροδέτης pin 5V. Ακροδέκτης σταθεροποιημένης τάσης 5Volt. Χρησιμοποιείται για την τροφοδοσία του μικροελεγκτή ή άλλων ηλεκτρονικών στοιχείων.
- Ακροδέκτης 3V3. Από αυτόν τον ακροδέκτη μέσω του ρυθμιστή που βρίσκεται πάνω στη πλατφόρμα μπορούμε να εξάγουμε 3.3 V.
- Ακροδέκτης GND. Ακροδέκτης Γείωσης.

ΜΝΗΜΗ

Η ATmega328 έχει 32 KB (0.5 KB καταλαμβάνεται από το bootloader). Διαθέτει επίσης 2 KB της SRAM και 1 KB EEPROM του (η οποία μπορεί να διαβάσει και να γράψει με τη βιβλιοθήκη EEPROM).

ΕΙΣΟΔΟΣ ΚΑΙ ΕΞΟΔΟΣ ΔΕΔΟΜΕΝΩΝ

Κάθε ένας από τους 14 ψηφιακούς ακροδέκτες πάνω στο Uno μπορεί να χρησιμοποιηθεί ως είσοδος ή έξοδος, χρησιμοποιώντας τις εντολές pinMode (), digitalWrite (), και τις λειτουργίες digitalRead (). Οι ακροδέκτες λειτουργούν στα 5V με δυνατότητα διαχείρισης 40 mA. Κάθε ακροδέκτης διαθέτει εσωτερικό αντιστάτη Pull-Up με τιμή αντίστασης 20-50 ΚΩ.

Σε αντίθεση, κάποια pins-ακροδέκτες έχουν κάποιες ειδικές λειτουργίες:

- Τα pin-ακδροδέκτες 0

- RX και TX χρησιμοποιούνται για να λαμβάνουν και να εκπέμπουν σειριακά δεδομένα. Αυτοί οι ακροδέκτες επικοινωνούν με τα αντίστοιχα pin του ATmega 8U2 και αυτός με τον υπολογιστή μέσω της USB θύρας.
- pins 2 και 3. Οι ακροδέκτες αυτοί χρησιμοποιούνται για την ενεργοποίηση διακοπών αν ανιχνευθεί παλμός χαμηλής τάσης. Η ενεργοποίηση των διακοπών πραγματοποιείται με την εφαρμογή της συνάρτησης attachInterrupt(). Η ενεργοποίηση των διακοπών μπορεί να γίνεται στο λογικό 0,1.
- pins 3, 5, 6, 9, 10, και 11. Λειτουργούν ως έξοδος 8-bit PWM με τη λειτουργία analogWrite () και με αυτό τον τρόπο δίνουν τη δυνατότητα να παρέχουν πληροφορίες για κάποια λειτουργία όπως πχ πόσες φορές ανοιγόκλεισε το φως σε κάποιο χρονικό διάστημα.
- Τα pin 10-SS, 11-MOSI, 12-MISO, 13-SCK υποστηρίζουν την επικοινωνία SPI, χρησιμοποιώντας τη βιβλιοθήκη SPI.
- Pin 13. Αν κοιτάξουμε προσεκτικά δίπλα στο pin 13 υπάρχει ένα led. Υπάρχει μία εσωτερική σύνδεση πάνω στη πλατφόρμα ανάμεσα σε αυτά τα δύο σημεία εξόδου από το pin 13 και εισόδου από το led. Δίνοντας τη κατάλληλη εντολή στο πρόγραμμα, το λαμπάκι αυτό ανάβει και επίσης δεν ανάβει.

Το Uno έχει 6 αναλογικές εισόδους, επισημαίνονται πάνω στη πλατφόρμα από A0 μέχρι A5, καθένας από τους οποίους έχει τη δυνατότητα να διαβάσει μια αναλογική τιμή και να τη μετατρέψει σε έναν αριθμό από 0 έως 1023. Επίσης έχει 14 ψηφιακούς ακροδέκτες από τους οποίους οι P3, P5, P6, P9, P10 και P11 έχουν την δυνατότητα να προγραμματιστούν ώστε να λειτουργούν και ως αναλογικές έξοδοι.

2.3 ETHERNET SHIELD

2.3.1 ΕΙΣΑΓΩΓΗ ΣΤΙΣ ΕΠΕΚΤΑΣΕΙΣ-SHIELDS

Επεκτάσεις ή αλλιώς shields όπως είναι γνωστές, είναι πλατφόρμες οι οποίες μπορούν να τοποθετηθούν στην πάνω μεριά του arduino, επεκτείνοντας τις δυνατότητες του. Υπάρχουν διαφορετικές shields με διαφορετικές λειτουργίες αλλά ακολουθούν την ίδια φιλοσοφία. Επίσης είναι εύκολες στο να τοποθετηθούν και δεν έχουν ιδιαίτερα υψηλό κόστος για να τις προμηθευτεί κάποιος σύμφωνα με τα σημερινά δεδομένα. Υπάρχει πολύ μεγάλος αριθμός διαφορετικών shield, επίσημων και ανεπίσημων, για αυτό το λόγο θα αναφέρω κάποιες από αυτές:

- Ethernet shield: Δίνει στο Arduino την δυνατότητα να συνδεθεί σε τοπικό δίκτυο ή στο διαδίκτυο μέσω ενός καλωδίου Ethernet.

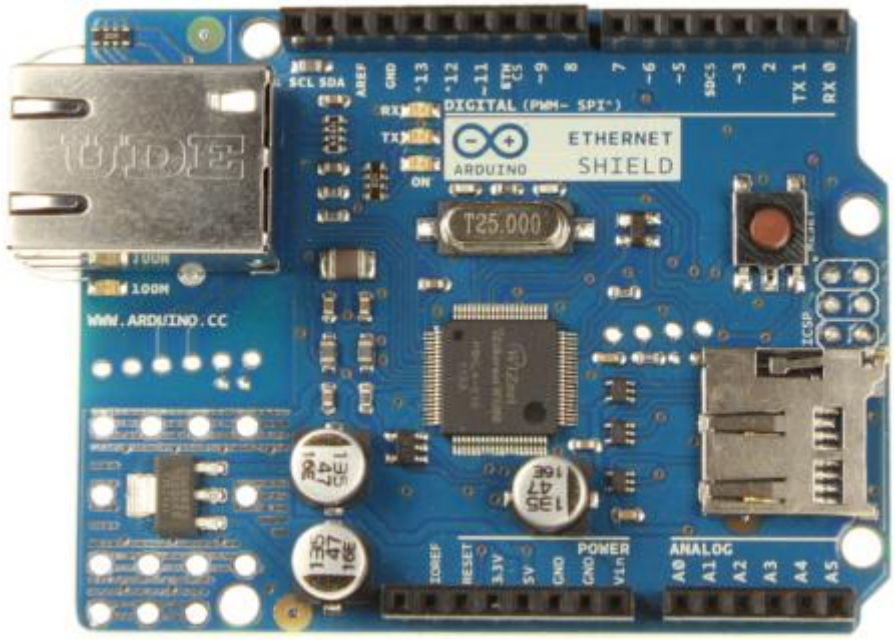
- WiFi shield: Δίνει στο Arduino την δυνατότητα να συνδεθεί σε τοπικό δίκτυο ή στο διαδίκτυο μέσω του ασύρματου δικτύου.
- Shield οθόνης: Προσθέτουν οθόνη στο Arduino.
- Wave shield: Δίνει στο Arduino την δυνατότητα να παίζει ήχους/μουσική από κάρτες SD.
- GPS shield: Δίνει τη δυνατότητα στο arduino να παρέχει τον εντοπισμό της θέσης που βρίσκεται μέσω του Global Positioning System (GPS, παγκόσμιο σύστημα εντοπισμού θέσης).
- Motor Shields: Δίνει την δυνατότητα στο arduino να διαχειριστεί κινητήρες (DC, servo κ.λπ.).
- ProtoShield: Μια προσχεδιασμένη πλακέτα πρωτοτυποποίησης, συμβατή στις διαστάσεις του Arduino και χωρίς εξαρτήματα για να κατασκευάζει ο χρήστης το δικό του shield.

Επίσης εκτός από τις shield υπάρχει μεγάλη ποικιλία εξαρτημάτων και αισθητήρων κίνησης, θερμοκράσιας, φωτός, καπνού, ήχου κι άλλα οι οποίοι έχουν συμβατότητα με τη πλατφόρμα toy arduino. Όλα αυτά μπορουν να χρησιμοποιηθούν σε πολλά κομμάτια της ζωής του ανθρώπου. «¨Εξυπνα σπίτια», αυτόματο πότισμα, αντικλεπτικά συστήματα μέχρι συστήματα διεύθυνσης και ρομποτική.

2.3.2 ETHERNET SHIELD

ΠΕΡΙΛΗΠΤΙΚΑ

Όπως αναφέρα στην προηγούμενη παράγραφο, το Arduino Ethernet Shield (ΕΙΚΟΝΑ 2) συνδέει την πλατφόρμα του Arduino στο διαδίκτυο. Για να λειτουργήσει, πρέπει να τη τοποθετήσουμε-συνδέσουμε πάνω στη βασική πλατφόρμα arduino και ύστερα μέσω ενός καλωδίου RJ45 στο δίκτυο που θέλουμε, στη συγκεκριμένη περίπτωση θα συνδεθεί με το modem. Επίσης βασική προϋπόθεση για να λειτουργήσει είναι να γνωρίζουμε την διεύθυνση MAC η οποία αναγράφεται πάνω σε ένα αυτοκόλλητο πάνω shield και θα χρειαστεί να την αναφέρουμε στα προγράμματα που θα παρουσιάσω σε επόμενη παράγραφο.



(Εικόνα 2. Ethernet shield)

Περιγραφή

Το Arduino Ethernet Shield Βασίζεται πάνω στο τσιπ ethernet Wiznet W5100 (φύλλο δεδομένων). Η Wiznet W5100 παρέχει ένα δίκτυο (IP) ικανό να λειτουργήσει στα πρότυπα TCP και UDP. Υποστηρίζει έως και τέσσερις ταυτόχρονες συνδέσεις. Η ethernet shield όπως θα παρατηρήσουμε στην από κάτω επιφάνεια της έχει κάποιες μεταλλικές κεφαλές οι οποίες κουμπώνουν-συνδέονται στους ακροδέκτες της πλακέτα Arduino.

Το Ethernet Shield έχει ένα πρότυπο RJ-45 σύνδεση, με ενσωματωμένο μετασχηματιστή γραμμής και Power over Ethernet ενεργοποιημένη.

Υπάρχει μια ενσωματωμένη υποδοχή κάρτας micro-SD, το οποίο μπορεί να χρησιμοποιηθεί για να αποθηκεύσει τα αρχεία για την εξυπηρέτηση μέσω του δικτύου. Είναι συμβατό με όλα τα μοντέλα Arduino / genuino. Τα δεδομένα της κάρτας micro SD είναι προσβάσιμα μέσω της Βιβλιοθήκης SD.

Στην επιφάνεια της υπάρχει ένας επανεκινητής πιο απλά ένα κουμπί RESET το οποίο είναι πολύ χρήσιμο για πρακτικούς λόγους καθώς επανεκινεί και τη λειτουργία της πλατφόρμας, όχι μόνο της Ethernet shield.Επίσης στην επιφάνεια της υπάρχουν τα εξής led :

PWR: υποδεικνύει ότι η πλατφόρμα του Arduino και το shield τροφοδοτούνται

LINK: δείχνει την ύπαρξη σύνδεσης σε κάποιο δίκτυο και αναβοσβήνει όταν το shield μεταδίδει ή λαμβάνει δεδομένα

FULLD: υποδεικνύει ότι η σύνδεση δικτύου είναι πλήρως αμφίδρομη

100M: ανάβει όταν το δίκτυο λειτουργεί στα 100 Mb / s (σε αντίθεση με τα 10 Mb / s)

RX: αναβοσβήνει όταν η shield λαμβάνει δεδομένα

TX: αναβοσβήνει όταν η shield στέλνει δεδομένα
COLL: αναβοσβήνει όταν ανιχνεύονται συγκρούσεις στο δίκτυο

2.4 ΣΥΝΔΕΣΗ ARDUINO UNO-ETHERNET SHIELD

Για να χρησιμοποιήσουμε την Ethernet shield, τη τοποθετούμε πάνω στη πλακέτα Arduino Uno. Για να ανεβάσουμε κάποιο πρόγραμμα στη πλατφόρμα όπως θα χρειαστεί αργότερα να κάνουμε, τη συνδέουμε στον υπολογιστή με ένα καλώδιο USB, όπως θα κάναμε και σε περίπτωση που δεν είχαμε σκοπό να χρησιμοποιήσουμε την Ethernet shield. Αφού το πρόγραμμα γραφτεί και φορτωθεί, τότε μπορούμε να αποσυνδέσουμε τη βασική πλατφόρμα UNO από τον υπολογιστή και να τη συνδέσουμε με ένα εξωτερικό τροφοδοτικό(Εικόνα 3).

(Εικόνα 3. Τοποθέτηση shield και arduino uno)

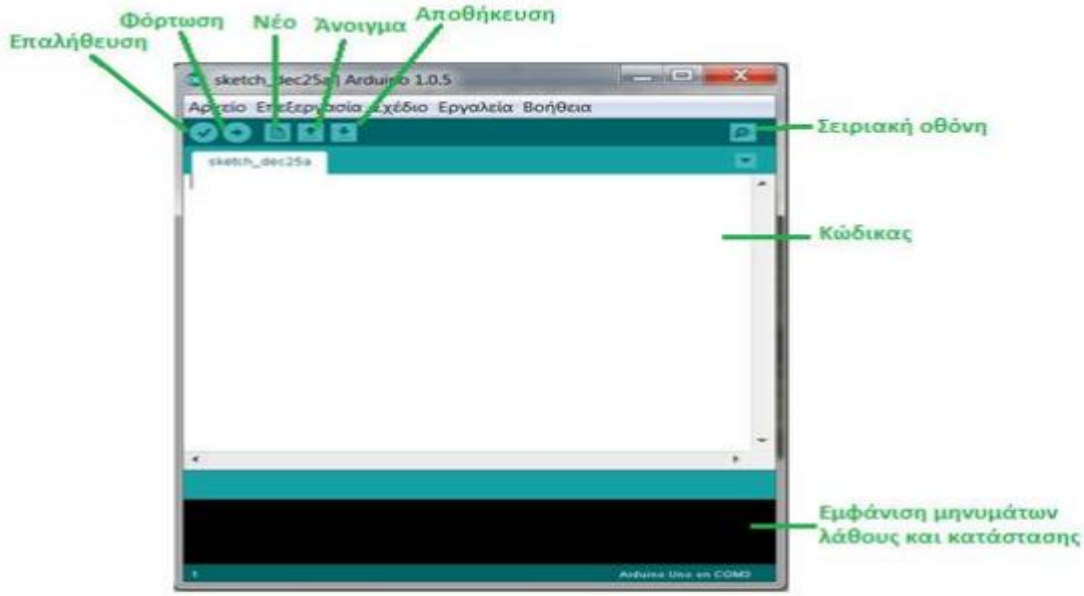
Η σύνδεση της shield πάνω στον υπολογιστή ή σε ένα διανομέα δικτύου ή δρομολογητή γίνεται χρησιμοποιώντας ένα τυπικό καλώδιο ethernet (CAT5 ή CAT6 με βύσματα RJ45). Είναι σημαντικό επίσης να αναφέρω ότι η σύνδεση σε υπολογιστή μπορεί να απαιτεί τη χρήση ενός καλωδίου cross-over (αν και πολλοί υπολογιστές, συμπεριλαμβανομένων όλων των πρόσφατων Macs μπορεί να κάνει το cross-over εσωτερικά).

Όσον αφορά τις ρυθμίσεις διαδικτύου που πρέπει να γίνουν, θα πρέπει για τη shield να υπάρχει μια διεύθυνση MAC και μια σταθερή διεύθυνση IP , καθώς και οι δύο διευθύνσεις θα δηλωθούν στο πρόγραμμα χρησιμοποιώντας τη συνάρτηση Ethernet.begin (). Η διεύθυνση MAC είναι ένα καθολικά μοναδικό αναγνωριστικό για μια συγκεκριμένη συσκευή. Η Ethernet έχει στη πάνω επιφάνεια της ένα αυτοκόλλητο που δείχνει τη διεύθυνση MAC. Σε περίπτωση που δεν υπάρχει αυτό το αυτοκόλλητο θα πρέπει να δημιουργήσουμε-χρησιμοποιήσουμε μία τυχαία και όχι την ίδια για κάθε μία αν οι πλατφόρμες που χρησιμοποιούμε στο σύστημα είναι παραπάνω από μία. Μια έγκυρη διεύθυνση IP εξαρτάται από τη διαμόρφωση του δικτύου μας.

Είναι δυνατή η χρήση DHCP για να εκχωρήσουμε δυναμικά την IP στην shield. Προαιρετικά, μπορούμε επίσης να καθορίσουμε μια πύλη δικτύου και υποδικτύου.

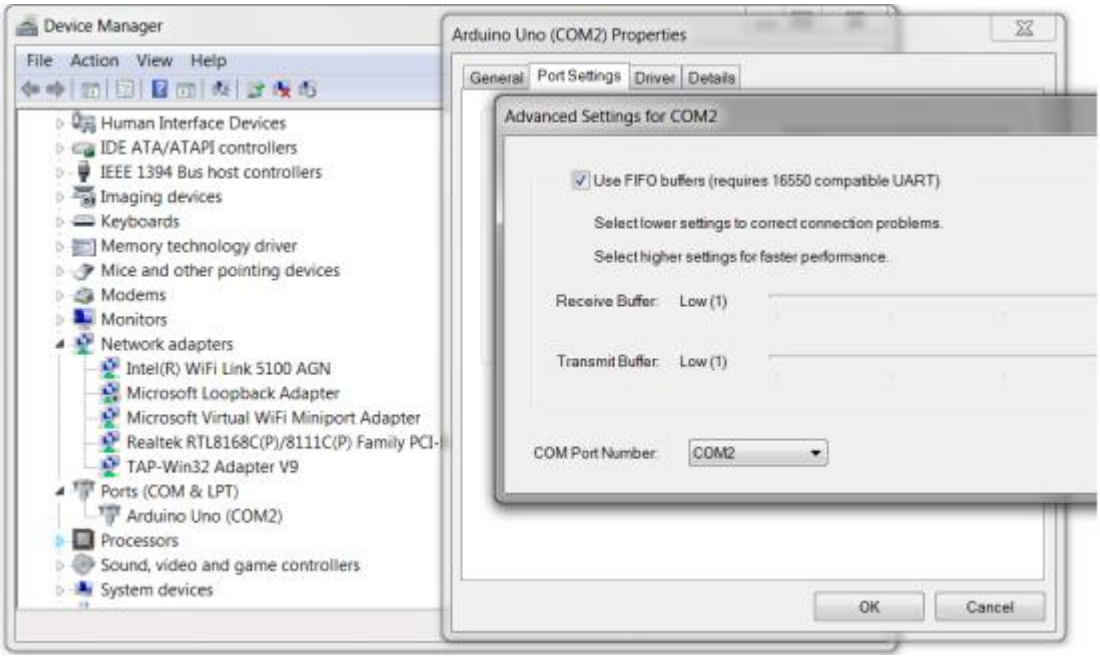
2.5 ΛΟΓΙΣΜΙΚΟ IDE

Η πλατφόρμα του arduino, όπως αναφέραμε έχει κάποιους ακροδέκτες, θύρες οι οποίες μπορούν να λειτουργήσουν είτε ως εισόδοι είτε ως εξόδοι. Η διαχείριση τους γίνεται μέσω προγραμμάτων, κώδικα τα οποία γράφουμε στο περιβάλλον προγραμματισμού IDE αφού πρώτα το κατεβάσουμε και το εγκαταστήσουμε στον υπολογιστή. Η πλατφόρμα του Arduino συνδέεται με τον υπολογιστή, μέσω της usb θύρας και μέσω αυτής της σύνδεσης κάνουμε upload τα προγράμματα στη πλατφόρμα.
Το IDE του Arduino χρησιμοποιεί το GNU toolchain και το AVR Libc για να μεταγλωττίζει προγράμματα και το avrdude για να φορτώνει προγράμματα στην πλακέτα. 21 Δεδομένου ότι η πλατφόρμα Arduino χρησιμοποιεί Atmel μικροελεγκτές, το περιβάλλον ανάπτυξης της Atmel, το AVR Studio ή το νεότερη έκδοση του Atmel Studio, μπορεί επίσης να χρησιμοποιηθεί για την ανάπτυξη λογισμικού για το Arduino.
Το λογισμικό IDE είναι διαθέσιμο για εκδόσεις Linux, Windows αλλά και Mac και μπορεί να γίνει download δωρεάν από την επίσημη ιστοσελίδα του Arduino , www.arduino.cc στην οποία υπάρχουν και οδηγίες για την εγκατάσταση του. Να αναφέρω επίσης, ότι το συγκεκριμένο περιβάλλον με τη κατάλληλη επιλογή μπορεί να εμφανίζεται και στα ελληνικά.
Στη παρακάτω εικόνα παρουσιάζεται η αρχική εικόνα(Εικόνα 3).

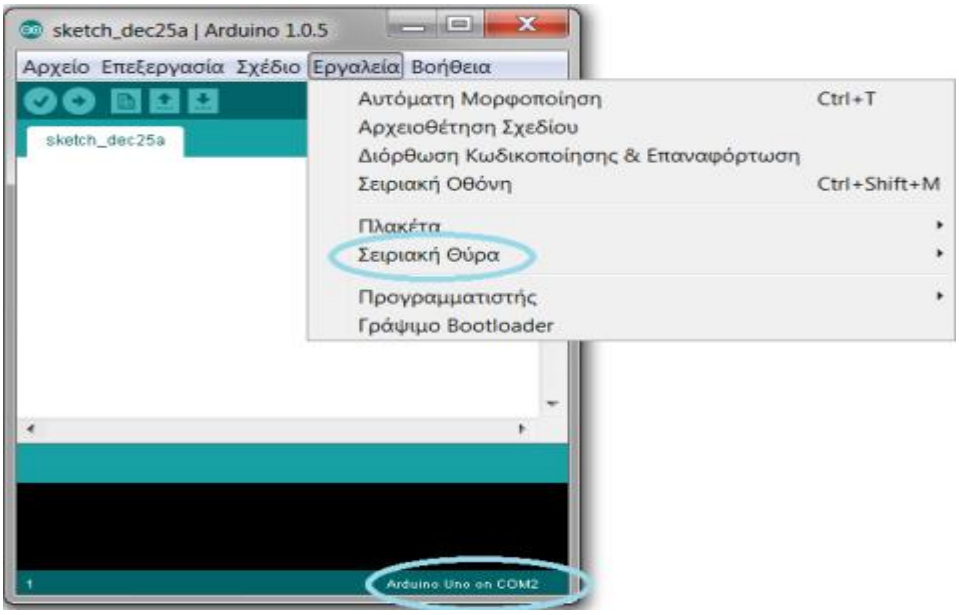


(Εικόνα 3,Αρχική εικόνα)

Κατά την εγκατάσταση του Arduino IDE πρέπει να ρυθμιστεί η σειριακή θύρα που επικοινωνεί το arduino με τον υπολογιστή. Στις δύο εικόνες που ακολουθούν παρατηρούμε δύο σημεία που πρέπει να ρυθμίσουμε την θύρα που δίνει ο υπολογιστής για τη σύνδεση και μετά τη θύρα που δίνει το λογισμικό (Εικόνα 4 και 5).



(Εικόνα 4. Ρύθμιση θύρας από control panel του υπολογιστή)



(Εικόνα 5. Ρύθμιση θύρας από το μενού του IDE)

ΚΕΦΑΛΑΙΟ 3-ANDROID

3.1 ΕΙΣΑΓΩΓΗ ΣΤΟ ANDROID

Πριν προχωρήσουμε στη σχεδίαση της εφαρμογής και παρουσίασης του κώδικα-υλοποίησης της εφαρμογής θα πρέπει να παρουσιάσω κάποια πληροφορίες για το Android. Το Anroid είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας-tablet, γενικά έξυπνες συσκευές, το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google. Το Android είναι κατά κύριο λόγο σχεδιασμένο για συσκευές με οθόνη αφής, όπως τα έξυπνα τηλέφωνα και τα τάμπλετ, με διαφορετικό περιβάλλον χρήσης για τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear). Παρόλο που έχει αναπτυχθεί για συσκευές με οθόνη αφής, έχει χρησιμοποιηθεί σε κονσόλες παιχνιδιών, ψηφιακές φωτογραφικές μηχανές, συνηθισμένους Η/Υ (π.χ. το HP Slate 21) και σε άλλες ηλεκτρονικές συσκευές.

Το Android είναι το πιο ευρέως διαδεδομένο λογισμικό στον κόσμο. Οι συσκευές με Android έχουν περισσότερες πωλήσεις από όλες τις συσκευές Windows, iOS και Mac OS X μαζί.

Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού. Το λογότυπο για το λειτουργικό σύστημα Android είναι ένα ρομπότ σε χρώμα πράσινου μήλου και σχεδιάστηκε από τη γραφίστρια Ιρίνα Μπλόκ (*Εικόνα* 6).

(Εικόνα 6. Λογότυπο Android)

3.2 ΙΣΤΟΡΙΑ ΕΚΔΟΣΕΩΝ

Η ιστορία εκδόσεων του Android του λειτουργικού συστήματος των κινητών ξεκίνησε με την κυκλοφορία του Android beta το Νοέμβριο του 2007. Η πρώτη εμπορική έκδοση ήταν το Android 1.0 που κυκλοφόρησε το Σεπτέμβριο του 2008. Το Android είναι υπό συνεχή ανάπτυξη από την Google και την Open Handset Alliance (OHA), και έχουν γίνει μια σειρά από ενημερώσεις στην λειτουργία του συστήματος από την αρχική κυκλοφορία του.

Από τον Απρίλιο του 2009, οι εκδόσεις του Android έχουν θέμα από την ζαχαροπλαστική στην κωδική ονομασία τους, και κυκλοφόρησαν σε αλφαβητική σειρά, εξαιρουμένων των εκδόσεων 1.0 και 1.1, που δεν τέθηκαν υπό συγκεκριμένα κωδικά ονόματα:

- Apple Pie (1.0)
- Banana Bread (1,1)
- Cupcake (1,5)
- Donut (1,6)
- Eclair (2.0-2.1)
- Froyo (2.2-2.2.3)
- Gingerbread (2.3-2.3.7)
- Honeycomb (3.0-3.2.6)

- Ice Cream Sandwich (4.0-4.0.4)
- Jelly Bean (4.1-4.3.1)
- KitKat (4.4-4.4.4)
- Lollipop (5.0-5.0.2)
- Marshmallow (6.0)

Μέσα στο 2016 αναμένεται νέα έκδοση του Marshmallow σύμφωνα με δημοσιεύματα.

3.3 ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ

Τα χαρακτηριστικά αναφέρονται στις τωρινές δυνατότητες των εκδόσεων του Android. Μέσα σε αυτά τα χαρακτηριστικά "κινείται" ο προγραμματιστής και είναι αυτά που παρουσιάζονται στο πίνακα 2:

Λειτουργίες Οθόνης	Η πλατφόρμα είναι προσαρμόσιμη σε πολλές ανάλυσεις οθόνης (από VGA μέχρι 4K), δισδιάστατες ψηφιακές γραφικές βιβλιοθήκες, τρισδιάστατα γραφικά βασισμένα στην OpenGL ES 3.0+ έκδοση χαρακτηριστικών, καθώς και παραδοσιακές απεικονίσεις οθόνης "έξυπνων" συσκευών κινητής τηλεφωνίας.
Αποθήκευση Δεδομένων	Χρήση βάσης δεδομένων SQLite για τις ανάγκες αποθήκευσης
Συνδεσιμότητα	Το Android υποστηρίζει τεχνολογίες συνδεσιμότητας συμπεριλαμβανομένου GSM/EDGE, 3G, 4G, CDMA,EV-DO, UMTS, Bluetooth, NFC, και Wi-Fi.
Αποστολή μηνυμάτων	SMS και MMS είναι οι διαθέσιμοι τρόποι ανταλλαγής μηνυμάτων.
Περιήγηση στον Ιστό	Για την περιήγηση στον ιστό το Android διαθέτει φυλλομετρητή βασισμένο στην ανοιχτή τεχνολογίαWebKit. Και άλλοι φυλλομετρητες είναι διαθέσιμοι από το Google play
Υποστήριξη	Λογισμικό γραμμένο στην Java είναι δυνατόν να μεταγλωττιστεί και να εκτελεστεί στην εικονική μηχανή Dalvik, η οποία αποτελεί

Java	εξειδικευμένη υλοποίηση εικονικής μηχανής, σχεδιασμένης για χρήση σε φορητές συσκευές, παρόλο που δεν είναι πρότυπη εικονική μηχανή Java.
Υποστήριξη Πολυμέσων	Το λειτουργικό Android υποστηρίζει τις ακόλουθα μορφές ήχου, στατικής και κινούμενης εικόνας: H.263,H.264 (σε 3GP ή MP4 container), MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, OGG Vorbis,WAV, JPEG, PNG, GIF, BMP.
Επιπλέον υποστήριξη υλικού	Το λειτουργικό Android μπορεί να συνεργαστεί με κάμερες στατικής ή κινούμενης εικόνας, οθόνες αφής,GPS, αισθητήρες επιτάχυνσης, μαγνητόμετρα, δισδιάστατους καθώς και τρισδιάστατους επιταχυντές γραφικών.
Περιβάλλον Ανάπτυξης Λογισμικού	Περιλαμβάνει ένας προσομοιωτή συσκευής, εργαλεία για διόρθωση σφαλμάτων, μνήμη και εργαλεία ανάλυσης της απόδοσης του εκτελέσιμου λογισμικού καθώς και ένα επιπρόσθετο για το Eclipse IDE.
Αγορά και Εγκατάσταση Εφαρμογών	Παρόμοια με το App Store του iPhone OS, το Google Play είναι ένας κατάλογος εφαρμογών που μπορούν να μεταφορτωθούν και εγκατασταθούν στην συσκευή άμεσα μέσω ασύρματων καναλιών, χωρίς την χρήση υπολογιστή. Αρχικά μόνο δωρεάν εφαρμογές ήταν δυνατόν να εγκατασταθούν. Εφαρμογές επί πληρωμή ήταν μετέπειτα διαθέσιμες στο Google play στις ΗΠΑ ύστερα από τις 19 Φεβρουαρίου 2009.
Οθόνη Αφής Πολλαπλών Σημείων	Το λειτουργικό Android υποστηρίζει οθόνες αφής πολλαπλών σημείων αλλά η δυνατότητα αυτή είχε κλειδωθεί σε επίπεδο πυρήνα (πιθανόν για αποφυγή παραβιάσεων των πατεντών λογισμικού της Apple στις τεχνολογίες οθονών αφής). Κυκλοφορούσε μια ανεπίσημη τροποποίηση (mod) που έχει αναπτυχθεί για να υποστηρίζει πολλαπλή επαφή (multi-touch), αλλά απαιτούσε δικαιώματα πρόσβασης υπερχρήστη (superuser) στη συσκευή για να γραφεί στη μνήμη flash ένας πυρήνας που να

	μην είναι υπογεγραμμένος (unsigned kernel). Από το Android 2.2 και ύστερα οι multi-touch displays έγιναν νόρμα.

(Πίνακας 2. Χαρακτηριστικά)

3.4 ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΩΝ ANDROID

Η ανάπτυξη λογισμικού Android είναι η διαδικασία με την οποία δημιουργούνται νέες εφαρμογές για το λειτουργικό σύστημα Android. Οι εφαρμογές αναπτύσσονται συνήθως στη γλώσσα προγραμματισμού Java, χρησιμοποιώντας το Android Software Development Kit ή το ANDROID STUDIO, αλλά και σε άλλα εργαλεία ανάπτυξης που είναι διαθέσιμα. Από τον Ιούλιο του 2013, περισσότερες από 1 εκατομμύριο εφαρμογές έχουν αναπτυχθεί για το Android, με πάνω από 25 δισεκατομμύρια downloads. Μια έρευνα τον Ιούνιο του 2011 έδειξε ότι πάνω από το 67% των προγραμματιστών που ανέπτυξαν εφαρμογές χρησιμοποίησε την πλατφόρμα. Στο δεύτερο τετράμηνο του 2012, περίπου 105 εκατομμύρια κινητά με λειτουργικό σύστημα Android πωλήθηκαν. Αυτό το γεγονός δίνει ένα συνολικό μερίδιο 68% στη συνολική πώληση smartphones μέχρι το 2ο τρίμηνο του 2012.

Στη δικιά μας περίπτωση(«ΕΞΥΠΝΟ ΣΠΙΤΙ», ΕΛΕΓΧΟΣ ΟΙΚΙΑΚΩΝ ΣΥΣΚΕΥΩΝ), εφαρμογή είναι το περιβάλλον το οποίο βασίζεται στο λειτουργικό σύστημα android και πάνω στο οποίο ο εκάστοτε χρήστης κάνει τη διαχείριση των συσκευών . Αυτό το περιβάλλον επικοινωνεί με τις οικιακές συσκευές μέσω του διαδικτύου στέλνοντας εντολές στον arduino ο οποίος τις εκτελεί.

3.5. ANDROID STUDIO

3.5.1 ΕΙΣΑΓΩΓΗ ΣΤΟ ANDROID STUDIO

Android Studio είναι το λογισμικό πάνω στο οποίο θα εργαστούμε για να αναπτύξουμε την εφαρμογή μας.
Android Studio είναι το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για την ανάπτυξη εφαρμογών Android. Το Android Studio προσφέρει ακόμα περισσότερα χαρακτηριστικά που ενισχύουν την παραγωγικότητα μας κατά τη δημιουργία Android εφαρμογών , όπως:

- Ένα ευέλικτο σύστημα κατασκευής Gradle –based
- Ένα γρήγορο και πλούσιο σε χαρακτηριστικά εξομοιωτή
- Ένα ενοποιημένο περιβάλλον , όπου μπορούμε να αναπτύξουμε για όλες τις συσκευές Android
- Άμεση Εκτέλεση για να ωθήσουμε τις αλλαγές στην λειτουργία της εφαρμογής μας , χωρίς την οικοδόμηση ενός νέου APK
- Κωδικός πρότυπα και την ολοκλήρωση GitHub για να μας βοηθήσει να οικοδομήσουμε κοινά χαρακτηριστικά app και δείγμα κώδικα εισαγωγής

- Εκτεταμένες εργαλεία ελέγχου και των πλαισίων εργαλεία χνούδι για να πιάσει την απόδοση , τη χρηστικότητα , την έκδοση συμβατότητας , και άλλα προβλήματα
- C ++ και υποστήριξη NDK
- Ενσωματωμένη υποστήριξη για την πλατφόρμα Google Cloud , καθιστώντας το εύκολο να ενσωματώσει το Google Cloud Messaging και App Engine

Θεωρώ ότι είναι σημαντικό να αναφέρω ότι κάθε προγραμματιστής σχεδιάζει και υλοποιεί κάποιο σχέδιο κάτω από τα δικά του πρότυπα, τρόπους σύμφωνα με το τι θεωρεί πιο εύκολο για αυτόν. Π.χ κάποιοι developer προτιμούν να να αναπτύσσουν τις εφαρμογές τους στο Eclipse το οποίο μαζί με όλο του το πακέτο (Java Development Kit,Android SDK, ADT Plug-in) για το Eclipse είναι και αυτό ένα εργαλείο για ανάπτυξη κώδικα. Στις μέρες μας οι πιο έμπειροι android developers δεν χρησιμοποιούν το Eclipse και γράφουν το κώδικα στο περιβάλλον του Android STUDIO. Θα προτιμήσω να αναπτύξω την εφαρμογή μου στο Android studio γιατί κατά τη γνώμη μου θεωρώ ότι είναι πιο εύκολη η έγκατάσταση του καθώς και πιο φιλικό το περιβάλλον του σε κάποιον που δεν είναι τόσο εξειδικευμένος.

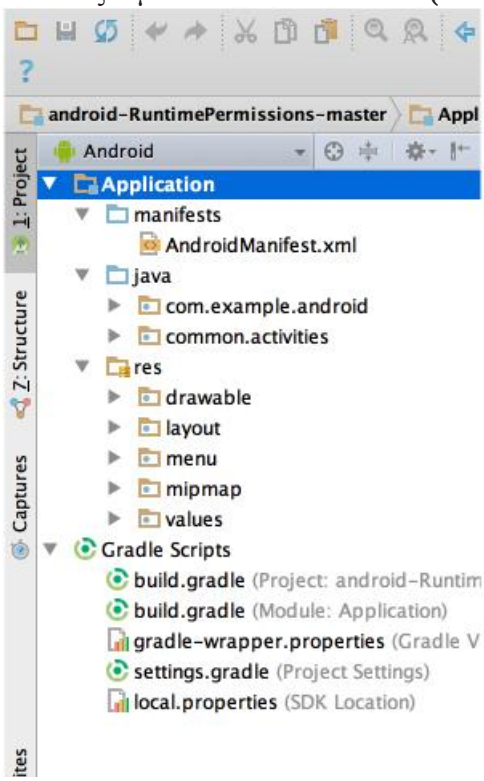
3.5.2 ΚΥΡΙΑ ΜΕΡΗ ANDROID STUDIO

Κάποια από τα μέρη που αποτελούν το λειτουργικό στο οποίο θα εργαστώ παραθέτονται παρακάτω.

-ΔΟΜΗ ΕΝΟΣ PROJECT

Κάθε έργο στο Android Studio περιέχει μία ή περισσότερες μονάδες με τα αρχεία πηγαίου κώδικα και τα αρχεία των πόρων.

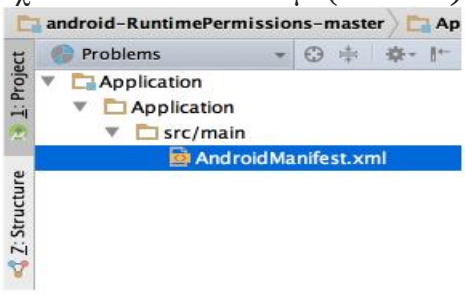
Από προεπιλογή, το Android Studio εμφανίζει τα αρχεία του project που θέλουμε να αναπτύξουμε στο Android View *(Εικόνα 7)*.



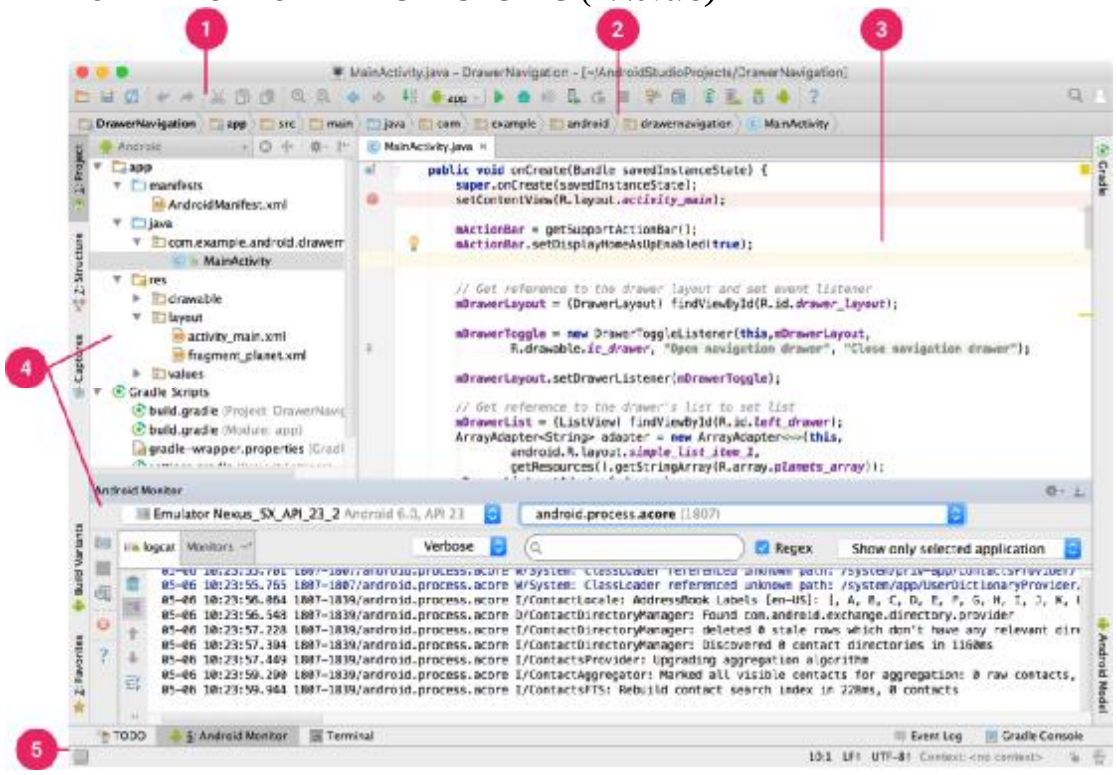
(Εικόνα 7. ANDROID VIEW)

Με αυτό το τρόπο μπορούμε να έχουμε γρήγορη πρόσβαση σε βασικά αρχεία προέλευσης του έργου μας. Όλα τα αρχεία κατασκευής είναι ορατά κάτω από το Gradle Scripts όπως βλέπουμε και στη παραπάνω εικόνα και κάθε project περιέχει τους εξής φακέλους:

manifests: Περιέχει το αρχείο AndroidManifest.xml.
java: Περιέχει τα αρχεία πηγαίου κώδικα Java, συμπεριλαμβανομένου του κώδικα δοκιμής JUnit.
res: Περιέχει όλους τους πόρους μη-κώδικα, όπως σχεδιαγράμματα XML, χορδές UI, και εικόνες bitmap.
Ωστόσο η δομή του έργου μπορεί να διαφέρει από αυτό το σχήμα. Για να δούμε την πραγματική δομή των αρχείων του project μας, επιλέξτε Project από το αναπτυσσόμενο μενού project όπως επίσης φαίνεται στη εικόνα 7.
Μπορούμε επίσης να προσαρμόσουμε την προβολή των αρχείων του έργου μας ώστε να επικεντρωθούμε και να βλέπουμε πιο ειδικά χαρακτηριστικά του project μας. Για παράδειγμα, επιλέγοντας την προβολή Προβλήματα του έργου μας, εμφανίζονται οι συνδέσεις με τα αρχεία προέλευσης που περιέχουν οποιαδήποτε αναγνωρισμένη κωδικοποίηση και τα συντακτικά λάθη, όπως κάποιο στοιχείο που μπορεί να λείπει, π.χ κάποιο XML κλείσιμο(*Εικόνα 8)*.



(Εικόνα 8. Εμφάνιση αρχείων που έχουν κάποιο πρόβλημα)
-ΚΥΡΙΟ ΠΑΡΑΘΥΡΟ ANDROID STUDIO (*Εικόνα 9)*



(*Εικόνα 8. Κεντρική εικόνα ANDROID STUDIO)*

1. Η γραμμή εργαλείων σας επιτρέπει να πραγματοποιήσουμε ένα ευρύ φάσμα δράσεων, συμπεριλαμβανομένης της λειτουργίας της εφαρμογής μας.
2. Η μπάρα πλοήγησης μας βοηθά να περιηγηθούμε μέσα από το έργο μας και να ανοίξουμε αρχεία για επεξεργασία. Παρέχει μια πιο συμπαγή άποψη της δομής ορατά στο παράθυρο του εργαλείου του έργου μας.
3. Το παράθυρο του επεξεργαστή είναι όπου μπορούμε να δημιουργήσουμε και να τροποποιήσουμε τον κώδικα. Ανάλογα με τον τρέχον τύπο αρχείου, το παράθυρο αυτό μπορεί να αλλάξει. Για παράδειγμα, κατά την προβολή ενός layout αρχείου , το παράθυρο του επεξεργαστή εμφανίζει τον layout editor και προσφέρει τη δυνατότητα να δούμε το αντίστοιχο αρχείο XML.
4. Τα εργαλεία παραθύρων (tool windows) μας δίνουν πρόσβαση σε συγκεκριμένες εργασίες, όπως τη διαχείριση του έργου, αναζήτηση, έλεγχος έκδοσης, και πολλά άλλα.
5. Η γραμμή κατάστασης εμφανίζει την κατάσταση του έργου μας και το ίδιο το IDE, καθώς και τυχόν προειδοποιήσεις ή μηνύματα.

Μπορούμε να οργανώσουμε το κύριο παράθυρο για να δώσουμε στον εαυτό σας περισσότερο χώρο στην οθόνη, αποκρύπτοντας ή μετακινώντας τις γραμμές εργαλείων και τα παράθυρα εργαλείων. Μπορούμε επίσης να χρησιμοποιήσουμε συντομεύσεις πληκτρολογίου για πρόσβαση στα περισσότερα χαρακτηριστικά IDE.

Ανά πάσα στιγμή, μπορούμε να κάνουμε αναζήτηση σε όλο τον πηγαίο κώδικα του project, βάσεις δεδομένων, τις δράσεις, τα στοιχεία της διεπαφής χρήστη, και ούτω καθεξής, με διπλό πάτημα του πλήκτρου Shift, ή κάνοντας κλικ στο μεγεθυντικό φακό στην επάνω δεξιά γωνία του Android Studio παράθυρο. Αυτό μπορεί να είναι πολύ χρήσιμο αν, για παράδειγμα, προσπαθούμε να εντοπίσουμε μια συγκεκριμένη ενέργεια IDE που έχετε ξεχάσει.

-ΧΕΙΡΙΣΜΟΣ
Το Android Studio έχει πάρα πολλές συντομεύσεις τις οποίες μπορούμε να χρησιμοποιούμε για να επιταχύνουμε διαδικασίες.
Εδώ είναι μερικές συμβουλές για να μας βοηθήσουν να κινηθούμε γύρω από το Android Studio.

- Control + E (Command + E σε Mac) για να εμφανιστεί η πρόσφατη δράση αρχείων. Από προεπιλογή, είναι επιλεγμένο το τελευταίο αρχείο που έγινε πρόσβαση. Μπορούμε επίσης να αποκτήσουμε πρόσβαση σε οποιοδήποτε παράθυρο του εργαλείου μέσω της αριστερής στήλης που αναφέραμε και παραπάνω.
- Μπορούμε να δούμε τη δομή του τρέχοντος αρχείου χρησιμοποιώντας την ενέργεια Δομή αρχείου-***File Structure*** action. Πατώντας Control + F12 (Command + F12 σε Mac). Χρησιμοποιώντας αυτήν την ενέργεια, μπορούμε γρήγορα να πλοηγηθούμε σε οποιοδήποτε μέρος του τρέχοντος αρχείου μας.
- Αν θέλουμε να αναζητήσουμε και να πλοηγηθούμε σε μια συγκεκριμένη κατηγορία στο έργο μας, χρησιμοποιούμε την Περιήγηση- *Navigate* στη δράση Class πατώντας Control + N (Command + O σε Mac).

- Μπορούμε να περιηγηθούμε σε ένα αρχείο ή φάκελο χρησιμοποιώντας την Περιήγηση-Navigate στη δράση αρχείο πατώντας Control + Shift + N (Command + Shift + O σε Mac).

Υπάρχουν και άλλες συντομεύσεις τις οποίες μπορεί να δεί εύκολα κάποιος στο site https://developer.android.com όπως και πολλά άλλα για το χειρισμό του ANDROID STUDIO.

3.5.3 ΕΓΚΑΤΑΣΤΑΣΗ ANDROID STUDIO

Το λειτουργικό μπορούμε να το κάνουμε download από το επίσημο site του android studio https://developer.android.com/studio/index.html.
Αφού ολοκληρωθεί το κατέβασμα-download ανοίγουμε τη γραμμή εντολών και πληκτρολογούμε "javac –version" για να δούμε αν έχουμε κάποιο JDK(Java Development Kit) εγκατεστημένο στον υπολογιστή μας. Αν η έκδοση που έχουμε δεν είναι κατάλληλη ή είναι μικρότερη από 1.8 κατεβάζουμε το JDK (Java Development Kit) από το http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html που να συμφωνεί με το λειτουργικό του υπολογιστή που εργαζόμαστε (*Εικόνα 9).*

(Εικόνα 9. http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.htm)

Υστερα τα βήματα για να κάνουμε install-εγκατάσταση σε λειτουργικό Windows του exe αρχείου που κάναμε download υπάρχουν στο https://developer.android.com/studio/install.html και παρουσιάζονται ως εξής:
«*Ξεκινήστε το .exe αρχείο που κατεβάσατε .*
Ακολουθήστε τον οδηγό εγκατάστασης για να εγκαταστήσετε το Android Studio και τυχόν απαραίτητα εργαλεία SDK .
Σε μερικά συστήματα Windows, το σενάριο εκτοξευτής δεν βρει όπου είναι εγκατεστημένο το JDK . Εάν αντιμετωπίσετε αυτό το πρόβλημα, θα πρέπει να ορίσετε μια μεταβλητή περιβάλλοντος που δείχνει τη σωστή θέση .
Επιλέξτε το μενού Έναρξη> Υπολογιστής > Ιδιότητες συστήματος> Advanced System Properties . Στη συνέχεια, ανοίξτε την καρτέλα Για προχωρημένους > Μεταβλητές Περιβάλλοντος και προσθέστε μια νέα μεταβλητή JAVA_HOME συστήματος που οδηγεί σε φάκελο JDK σας , για παράδειγμα, C: \ Program Files \ Java \ jdk1.8.0_77» .

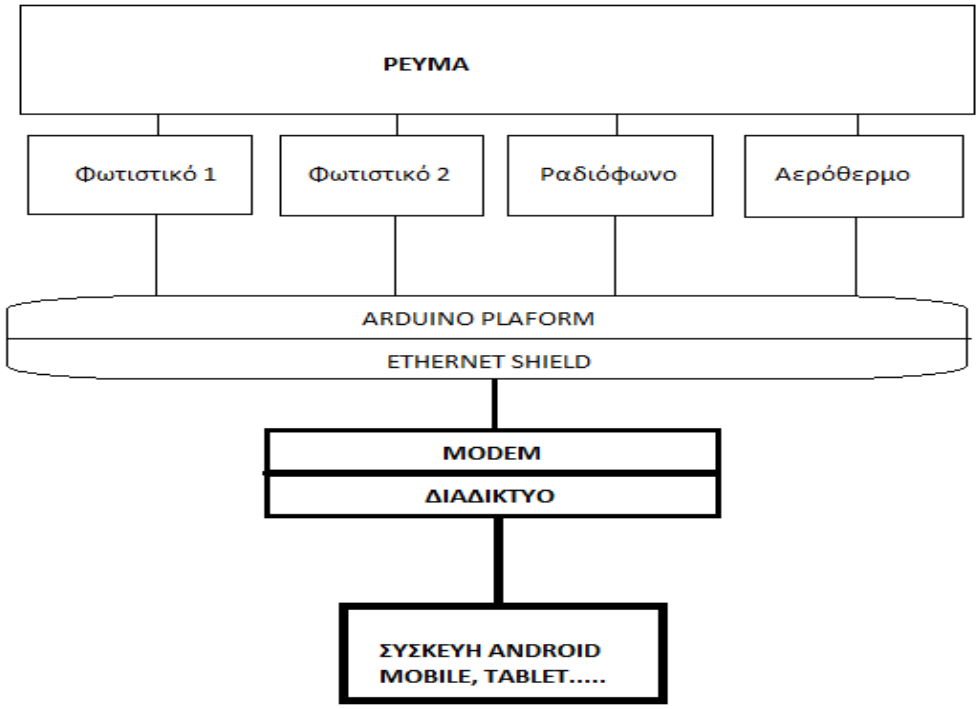
ΚΕΦΑΛΑΙΟ 4 ΣΧΕΔΙΑΣΗ ΣΥΣΤΗΜΑΤΟΣ ΒΗΜΑ-ΒΗΜΑ

4.1 ΑΝΑΚΕΦΑΛΑΙΩΣΗ

Ανακεφαλαιώνοντας, έχουμε μελετήσει τη πλατφόρμα του Arduino και την Ethernet SHIELD-επέκταση του, έχω αναφέρει κάποια πράγματα για το λογισμικό ARDUINO IDE στο ίδιο κεφάλαιο με τα προηγούμενα δύο και έχω επίσης ήδη αναφερθεί στο περιβάλλον ANDROID. Να σημειώσω ότι ακολουθώντας την ανάγνωση του κειμένου από την αρχή προς το τέλος, δεν έχω αναφερθεί ακόμα στις γλώσσες προγραμματισμού και στους κώδικες των προγραμμάτων και ο λόγος είναι ότι προσπαθώ να ακολουθήσω ένα καθοδηγητικού χαρακτήρα ύφος αναφέροντας τα βήματα όπως έγιναν και στη πραγματικότητα**.**

4.2 ΓΕΝΙΚΗ ΑΠΟΨΗ ΣΧΕΔΙΟΥ

Έχοντας αποφασίσει ότι οι οικιακές συσκευές θα είναι δυο φωτιστικά γραφείου, το ραδιόφωνο και το αερόθερμο και ότι αυτές θα ελέγχονται μέσω της android εφαρμογής που αφού επίσης σχεδιάσω ύστερα θα την αναπτύξω και θα την υλοποιήσω, έφτιαξα αρχικά ένα απλό σχέδιο που δείχνει μια γενική άποψη.



(Εικόνα 10. Γενική άποψη του σχεδίου)

Ωστόσο το σχέδιο μας δεν είναι τόσο απλό όσο φαίνεται στο παραπάνω σχέδιο. Για να γίνει πιο κατανοητό το τι θέλουμε να κάνουμε και να εξελίξουμε το παραπάνω

σχέδιο, θα πρέπει σε αυτό το σημείο να περιγράψω για ποιο λόγο και με ποιο τρόπο συνδέεται η πλατφόρμα του arduino με τις τέσσερις συσκευές.

Η πλατφόρμα του arduino συνδέεται με τις συσκευές γιατί είναι η υπεύθυνη συσκευή για τον έλεγχο τους. Πιο συγκεκριμένα, η πλατφόρμα Ethernet shield όπως είπαμε τοποθετείται πάνω στη πλατφόρμα πάνω στους ακροδέκτες της πλατφόρμας. Στο πάνω μέρος της η Ethernet shield έχει επίσης κάποιους ακροδέκτες, οι οποίοι λειτουργούν ακριβώς όπως και οι ακροδέκτες του arduino. Σκοπός μας είναι μέσω αυτών των ακροδεκτών να ασκήσουμε τον έλεγχο στις ηλεκτρικές συσκευές. Με ποιο τρόπο θα γίνει όμως αυτό?

Απαντώντας στο ερώτημα πως γίνεται ο έλεγχος των συσκευών από τη platforma, την απάντηση μας την έδωσε μια άλλη συσκευή πολύ μικρή, ένα ανταλλακτικό θα λέγαμε πιο απλά με την ονομασία relay και στα Ελληνικά Ηλεκτρονόμος. Στην επόμενη ενότητα θα αναφερθώ στους ηλεκτρονόμους.

4.3 ΗΛΕΚΤΡΟΝΟΜΟΣ

Ο **ηλεκτρονόμος**, **ρελέ** (*relay*) ή **ρελές** είναι ένας ηλεκτρικός διακόπτης που ανοίγει και κλείνει ένα ηλεκτρικό κύκλωμα κάτω από τον έλεγχο ενός άλλου ηλεκτρικού κυκλώματος. Στην αρχική μορφή του, ένας ηλεκτρομαγνήτης ενεργοποιούσε το διακόπτη, με το άνοιγμα ή κλείσιμο μιας ή περισσότερων επαφών. Εφευρέθηκε από τον Τζόζεφ Χένρυ το 1835. Επειδή ένας ηλεκτρονόμος είναι ικανός να ελέγχει ένα κύκλωμα εξόδου υψηλότερης ισχύος από το κύκλωμα εισόδου, μπορεί να θεωρηθεί, γενικά, μια μορφή ηλεκτρικού ενισχυτή.

Κάθε επαφή ενός ηλεκτρονόμου μπορεί να είναι *Κανονικά-Ανοικτή* (*Normally Open, NO*), *Κανονικά-Κλειστή'* (*Normally Closed, NC*) ή *μεταγωγικός* (*change-over*), ανάλογα με τον τύπο της.

- Μια επαφή **Κανονικά-Ανοικτή** συνδέει το κύκλωμα όταν ο ηλεκτρονόμος ενεργοποιείται· το κύκλωμα αποσυνδέεται όταν ο ηλεκτρονόμος είναι ανενεργός. Μια τέτοια επαφή καλείται επίσης *Επαφή Μορφής Α* ή *επαφή "make"*. Η επαφή μορφής Α είναι ιδανική για εφαρμογές που απαιτούν την ενεργοποίηση μιας πηγής υψηλής τάσης από απόσταση.
- Μια επαφή **Κανονικά-Κλειστή** αποσυνδέει το κύκλωμα όταν ο ηλεκτρονόμος ενεργοποιείται· το κύκλωμα συνδέεται όταν ο ηλεκτρονόμος είναι ανενεργός.

Μια τέτοια επαφή καλείται επίσης *Επαφή Μορφής Β* ή *επαφή "break"*. Η επαφή μορφής Β είναι ιδανική για εφαρμογές που απαιτούν το κύκλωμα να παραμένει κλειστό (ενεργό) μέχρι ο ηλεκτρονόμος να ενεργοποιηθεί.

- Μια επαφή **Μεταγωγική** μπορεί να ελέγχει δύο κυκλώματα. Ισοδυναμεί με μια επαφή κανονικά-ανοικτή και μια επαφή κανονικά-κλειστή που έχουν ένα κοινό ακροδέκτη. Μια τέτοια επαφή καλείται επίσης *Επαφή Μορφής C*.

Συνήθως ένας ηλεκτρονόμος αποτελείται από περισσότερες από μία ελεγχόμενες επαφές. Οι επαφές χωρίζονται σε κύριες και βοηθητικές. Οι κύριες διαρρέονται συνχά από ισχυρότερα ρεύματα και έτσι είναι αυτές που διακόπτουν το κύριο κυκλωμα και συνήθως είναι **Κανονικά-Ανοικτές**. Οι βοηθητικές έχουν όπως υποννοεί και το όνομά τους επικουρικό χαρακτήρα και ο ρόλος τους είναι να βοηθούν στον έλεγχο των αυτοματισμών (που είναι ο κύριος τομέας χρήσης των ηλεκτρονόμων). Για παράδειγμα βοηθούν στην ενεργοποίηση/απενεργοποίηση βοηθητικών κυκλωμάτων όπως ενδεικτικές λυχνίες.

ΛΕΙΤΟΥΡΓΙΑ

(Εικόνα 11. Ένας ηλεκτρονόμος στερεάς κατάστασης, ο οποίος δεν έχει κινούμενα μέρη)

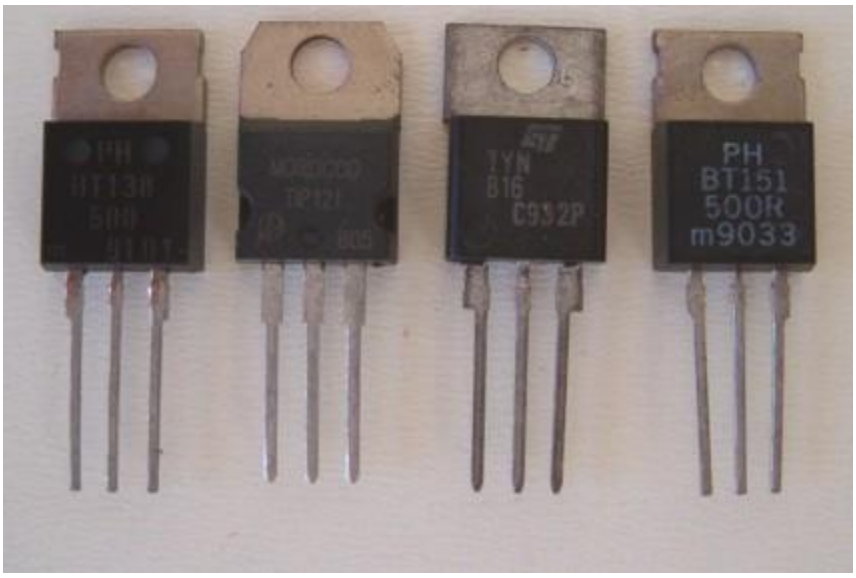
Όταν ηλεκτρικό ρεύμα διαρρέει το πηνίο του ηλεκτρονόμου, το παραγόμενο μαγνητικό πεδίο έλκει έναν οπλισμό που είναι μηχανικά συνδεδεμένος σε μια κινούμενη επαφή. Έτσι, η κινούμενη επαφή είτε συνδέεται με μια σταθερή επαφή είτε αποσυνδέεται από τη σταθερή επαφή. Μόλις το ηλεκτρικό ρεύμα στο πηνίο διακοπεί, ο οπλισμός επιστέφει στη θέση ηρεμίας του εξαιτίας μιας δύναμης επαναφοράς, που είναι ίση με το ήμισυ της μαγνητικής. Η δύναμη επαναφοράς παρέχεται συνήθως από ένα ελατήριο, αλλά και η βαρύτητα χρησιμοποιείται συχνά σε βιομηχανικούς εκκινητές μηχανών. Η μεταβολή τησμαγνητικής ροής στο πηνίο γεννά ένα ηλεκτρικό ρεύμα, το λεγόμενο "επαγωγικό", που έχει αντίθετη φορά από εκείνο που παρέχεται στο πηνίο. Για τη λειτουργία του πηνίου και τη μετακίνηση των επαφών απαιτείται σχετικά μεγάλη ένταση ηλεκτρικού ρεύματος, αλλά - μόλις ο οπλισμός κλείσει - το ηλεκτρικό ρεύμα που απαιτείται για να κρατήσει τον οπλισμό κλειστό είναι ένα μικρό κλάσμα του αρχικού, τυπικά το $^{1}/_{10}$. Οι ηλεκτρονόμοι κατασκευάζονται για να λειτουργούν γρήγορα. Σε μια εφαρμογή χαμηλής τάσης,

αυτό γίνεται για τη μείωση του θορύβου. Σε μια εφαρμογή υψηλής τάσης ή υψηλής έντασης ρεύματος, αυτό γίνεται για τη μείωση
των σπινθηρισμών (ηλεκτρικών εκφορτίσεων μορφής τόξου).

Εάν το πηνίο διεγείρεται με συνεχές (DC) ρεύμα, ανεξάρτητα από το ηλεκτρικό ρεύμα που ρέει διαμέσου των επαφών, μια δίοδος μπαίνει συνήθως παράλληλα με το πηνίο. Όταν το πηνίο διεγείρεται, αποκαθίσταται ένα μαγνητικό πεδίο. Όταν το πηνίο αποδιεγείρεται, το καταρρέον μαγνητικό πεδίο δημιουργεί μια αιχμή ηλεκτρικού ρεύματος που θα μπορούσε να βλάψει το υπόλοιπο κύκλωμα. Αν το πηνίο διεγείρεται με εναλλασσόμενο (AC) ρεύμα, ένα μικρό χάλκινο δαχτυλίδι πτυχώνεται στο άκρο του σωληνοειδούς πηνίου. Το εναλλασσόμενο ρεύμα μηδενίζεται 100 φορές το δευτερόλεπτο. Σε κάθε χρονική στιγμή μηδενισμού, δεν υπάρχει καμιά μαγνητική δύναμη που να συγκρατεί τις επαφές κλειστές. Το μικρό χάλκινο δαχτυλίδι παρέχει ένα μικρό ρεύμα εκτός φάσεως που καλείται ***shadow pole*** *(σκιώδης πόλος)*. Το άθροισμα του εναλλασομένου ρεύματος και του ***shadow pole*** εξασφαλίζει τη συγκράτηση του οπλισμού στη θέση εμπλοκής σε όλες τις χρονικές στιγμές.

4.4 ΤΡΑΝΖΙΣΤΟΡ

Για τη λειτουργία του σχεδίου μας θα χρειαστούμε και μια άλλη μικρή συσκευή η οποία αν και είναι μικρή σε μέγεθος, μικρότερη από ένα relay, είναι πολύ σημαντική και απαραίτητη. Αυτή η συσκευή είναι το τρανζίστορ (*εικόνα 12).*



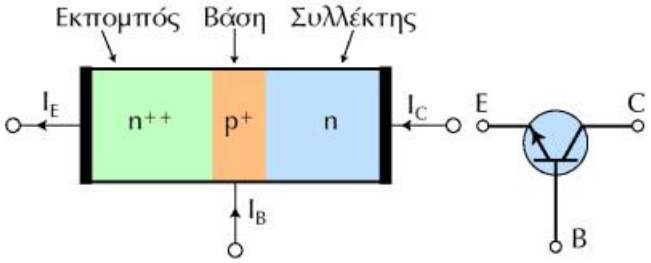
(Εικόνα 12. Τρανζίστορ)

Το τρανζίστορ (αγγλικά: transistor), στα ελληνικά κρυσταλλοτρίοδος και παλαιότερα κρυσταλλολυχνία, είναι διάταξη ημιαγωγών στερεάς κατάστασης, η οποία βρίσκει διάφορες εφαρμογές στην ηλεκτρονική, μερικές εκ των οποίων είναι η ενίσχυση, η σταθεροποίηση τάσης, η διαμόρφωση συχνότητας, η λειτουργία ως διακόπτης και ως μεταβλητή ωμική αντίσταση. Το τρανζίστορ μπορεί, ανάλογα με

την τάση με την οποία πολώνεται, να ρυθμίζει την ροή του ηλεκτρικού ρεύματος που απορροφά από συνδεδεμένη πηγή τάσης. Τα τρανζίστορ κατασκευάζονται είτε ως ξεχωριστά ηλεκτρονικά εξαρτήματα είτε ως τμήματα κάποιου ολοκληρωμένου κυκλώματος.

Το τρανζίστορ θεωρείται μία από τις μεγαλύτερες εφευρέσεις του 20ου αιώνα. Είναι το κυριότερο συστατικό όλων σχεδόν των σύγχρονων ηλεκτρονικών κατασκευών. Η πλατιά χρήση του οφείλεται κυρίως στη δυνατότητα παραγωγής του σε τεράστιες ποσότητες που μειώνουν το κόστος ανά μονάδα. Παρόλο που αρκετοί παραγωγοί παράγουν, ακόμα και σήμερα, μεμονωμένες συσκευασίες τρανζίστορ, η μεγαλύτερη ποσότητα παράγεται μέσα σε ολοκληρωμένα κυκλώματα (που συχνά αναφέρονται ως τσιπς) μαζί με τις διόδους, αντιστάσεις, πυκνωτές και άλλα ηλεκτρονικά εξαρτήματα.

Ο τρόπος λειτουργίας του έχει να κάνει με τις τρεις ακίδες που το αποτελούν εκπομπό e, βάση b και συλλέκτη c (εικόνα 13).

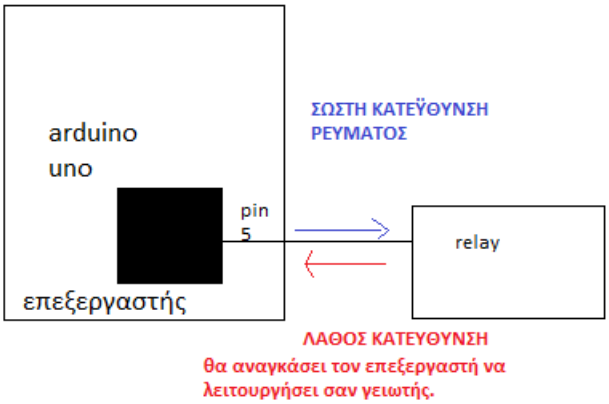


(Εικόνα 13. e , b και c)

Θεωρώ στο συγκεκριμένο σημείο ανούσιο να παρουσιάσω περισσότερες πληροφορίες για τα τρανζίστορ καθώς υπάρχουν πάρα πολλά κυκλώματα στα οποία μπορούν να χρησιμοποιηθούν και αυτό που μας ενδιαφέρει περισσότερο είναι ο λόγος χρήσης τους και ο τρόπος λειτουργίας τους στο σχέδιο μας. Η χρήση του τρανίνστορ στο σχέδιο μας έχει να κάνει με τη σωστή λειτουργία του κυκλώματος μας.

Όπως μάθαμε στα προηγούμενα κεφάλαια τα pins του arduino παράγουν ρεύμα από τη καρδιά της πλατφόρμας τον μικροεπεξεργαστή. Με τη χρήση του transistor θα αποφύγουμε την εναλλαγή του ρεύματος στη κατεύθυνση του. Πιο απλά θα αποφευχθεί ο κίνδυνος όπως φεύγει το ρεύμα από τον μικροεπεξεργαστή και βγαίνει από τα pins να αντιστραφεί η κατεύθυνση του προς την αντίθετη κατεύθυνση και να

αναγκαστεί ο επεξεργαστής να λειτουργήσει ως γείωση (Εικόνα 14).



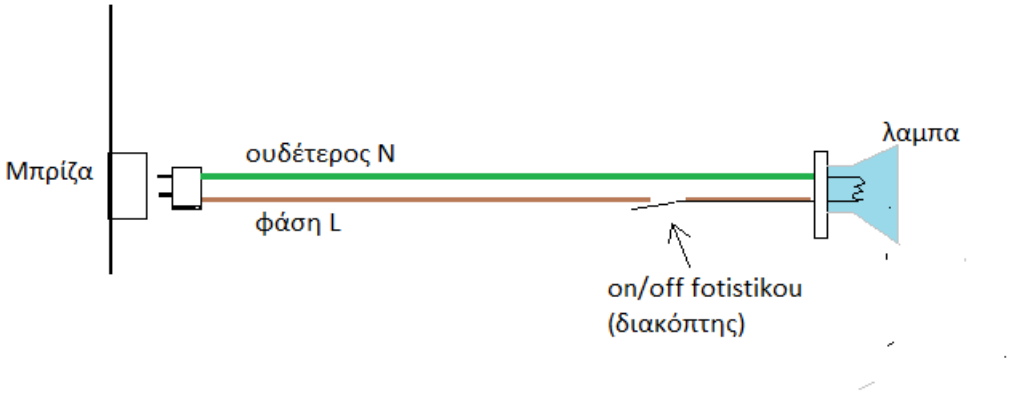
(Εικόνα 14. Εναλλαγή ρεύματος)

Επίσης κάτι τέτοιο θα αναγκάσει το σύστημα να μη λειτουργεί σωστά και να μη γνωρίζουμε το λόγο καθώς μπορεί να νομίζουμε ότι η πλατφόρμα στέλνει ρεύμα ενώ αυτό γυρνάει στον επεξεργαστή και να μη γνωρίζουμε το λόγο που αυτό δεν λειτουργεί. Με τη χρήση του transistor είμαστε πιο σίγουροι ότι το κύκλωμα λειτουργεί σωστά. Στο επόμενο κεφάλαιο εικόνα τάδε παρουσιάζεται ο τρόπος σύνδεσης του.

4.5 ΣΥΝΔΕΣΗ ARDUINO-ΤΡΑΝΖΙΣΤΟΡ-ΗΛΕΚΤΡΟΝΟΜΟΣ-ΣΥΣΚΕΥΕΣ

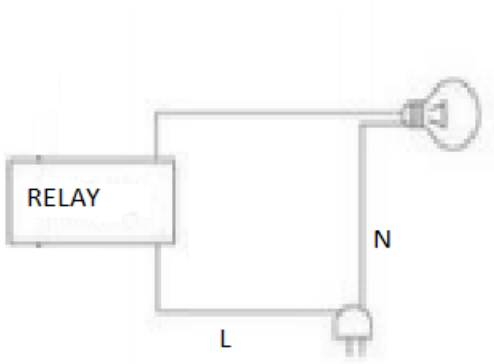
4.5.1 ΚΥΚΛΩΜΑΤΑ

Με πιο απλά λόγια, ο ηλεκτρονόμος είναι η συσκευή όπου θα μας βοηθήσει να ασκήσουμε αυτοματισμούς στις λειτουργιές ON και OFF για τις συσκευές μας. Στην εικόνα 15 βλέπουμε ένα σχέδιο κυκλώματος για το πως δουλεύει η συσκευή του φωτιστικού.



(Εικόνα 15. Κύκλωμα τροφοδοσίας φωτιστικού)

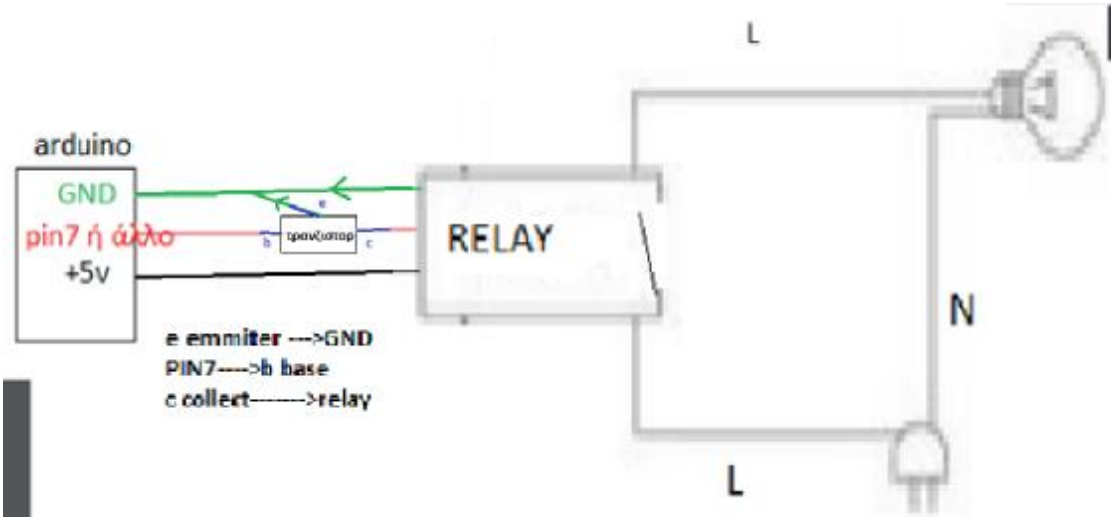
Με τον ίδιο τρόπο δουλεύει και το κύκλωμα των υπολοίπων 3ών συσκευών που θα χρησιμοποιήσουμε. Θεωρόντας τη χρήση του relay το σχέδιο του κυκλώματος για το φωτιστικό θα γίνει όπως στην παρακάτω εικόνα(16):



(*Εικόνα 16. Σύνδεση Φωτιστικού με relay)*

Δηλαδή όπως βλέπουμε και στην εικόνα στην κάθε συσκευή θα πρέπει να γίνει κάποια μικρή παραμετροποίηση, καθώς στο καλώδιο της φάσης (σε όλες τις συσκευές από πιστοποιημένα εργοστάσια είναι το μαύρο καλώδιο) μεσολαβεί το relay.

Στην επόμενη εικόνα(17) βλέπουμε το σχέδιο βάζοντας και τη συσκευή του arduino.



(*Εικόνα 17. Σύνδεση Arduno(Μέσω της Ethernet shield-relay-φωτιστικό)*

4.5.2 ΕΠΙΛΟΓΗ ΚΑΙ ΣΥΝΔΕΣΗ ΣΤΟΙΧΕΙΩΝ-ΤΥΠΟΙ ΚΑΛΩΔΩΝ-ΠΑΡΑΜΕΤΡΟΠΟΙΗΣΗ ΣΥΣΚΕΥΩΝ

Για το σχέδιο μου επέλεξα να χρησιμοποιήσω ένα relay module με 4 chanel όπως αυτό που βλέπετε στην εικόνα 18.

(Εικόνα 18. Relay Module 4 chanel-με 4 κανάλια)

Θα ήθελα να αναφέρω ότι υπάρχει επίσης η δυνατότητα να χρησιμοποιήσουμε και 4 ξεχωριστά relays με ένα κανάλι το καθένα για κάθε συσκευή όπως αυτό που βλέπετε στην εικόνα 19. Κατά τη γνώμη μου θεωρώ ότι για το στόχο μας το ένα relay με τα 4 chanel είναι μια καλή επιλογή και εύχρηστη για το σχέδιο μας.

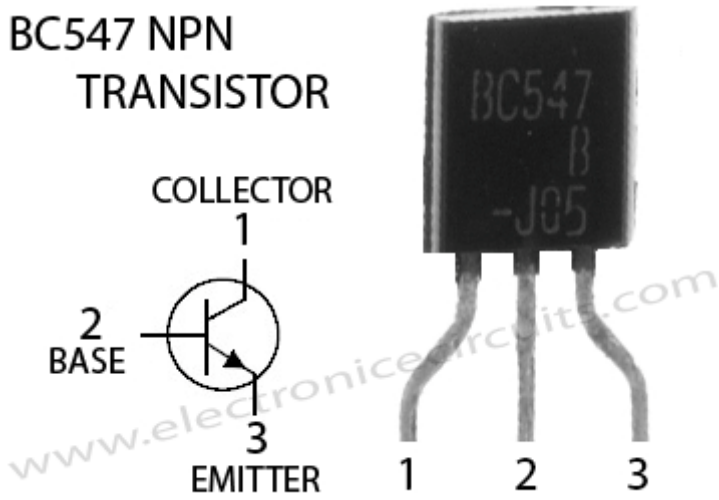
(Εικόνα 19. Relay Module 1 chanel)

Για τη σύνδεση του relay module με τα τρανζίστορ και τους ακροδέκτες της πλατφόρμας μας χρησιμοποιήσαμε καλώδια jumper χρωμάτων διαφόρων χρωμάτων male to female (*Εικόνα 20*) και male to male(*Εικόνα 21*). Η πλευρά των female συνδέεται στη μία πλευρά του relay καθώς η άλλη χρησιμοποιείται για να συνδεθούν οι συσκευές.

(*Εικόνα 20. Καλώδια jumper male to female)*

(Εικόνα 21. Καλώδια jumper Male to Male)

Πιο συγκεκριμένα στην Ethernet shield θα χρησιμοποιήσουμε τα digital pins 6, 7 , 8 , 9 που είναι τα αντίστοιχα της πλατφόρμας του arduino και τα οποία λειτουργούν ως trigger ξεχωριστά το καθένα για κάθε συσκευή, το gnd ως γείωση και το 5V ως τάση για τη λειτουργία του relay module. Θα παρεμβάλουμε όπως δείξαμε και σε προηγούμενο σχέδιο μεταξύ των συνδέσεων τα τρανζίστορ. Τα τρανζίστορ που θα χρησιμοποιήσουμε είναι το BC547 (Εικόνα 22) και η ποσότητα που απαιτεί το σύστημα μας είναι 5.



(*Εικόνα 22. Τρανζίστορ BC547)*

Οι Συνδέσεις έχουν ως εξής:

ΜΑΥΡΟ ΚΑΛΩΔΙΟ
[Arduino uno] 5v à b base[transistor 1]
c collect [transistor 1] à VCC [relay]
e emmiter [transistor 1] à GND [Arduino uno]
ΚΟΚΚΙΝΟ ΚΑΛΩΔΙΟ
[Arduino uno]pin6 à b base [transistor 2]
C collect [transistor 2] à int 1[relay]
e emmiter [transistor 2] à GND [Arduino uno]
ΠΡΑΣΙΝΟ ΚΑΛΩΔΙΟ
[Arduino uno]pin7 à b base [transistor 3]
C collect [transistor 3] à int 2[relay]
e emmiter [transistor 3] à GND [Arduino uno]
ΜΠΛΕ ΚΑΛΩΔΙΟ
[Arduino uno]pin8 à b base [transistor 4]
C collect [transistor 4] à int 3[relay]
e emmiter [transistor 4] à GND[Arduino uno]

ΜΑΥΡΟ ΚΑΛΩΔΙΟ
[Arduino uno]pin9 à b base [transistor 5]
C collect [transistor 5] à int 1[relay]
e emmiter [transistor 5] à GND [Arduino uno]
ΠΡΑΣΙΝΟ ΚΑΛΩΔΙΟ
[relay] GND à GND [Arduino uno]

(Οι συνδέσεις που οδηγούν στο GND του ARDUINO μπορούν να οδηγηθούν και σε κάποια εξωτερική γείωση (Ως γείωση μπορούμε να χρησιμοποιήσουμε το οποιοδήποτε μέταλλο) ή τα καλώδια που πρόκειται να συνδεθούν στο GND του ARDUINO κατά τα διαδρομή τους προς αυτό να οδηγηθούν σε ένα καλώδιο και μετά αυτό το καλώδιο στο GND του arduino.) ***Σημειώνεται ότι μπορεί να χρησιμοποιηθούν και διαφορετικά χρώματα καλωδίων-αναφέρονται απλά ως πρόταση)***

Όσον αφορά την άλλη πλευρά που συνδέεται με τις συσκευές, βλέπουμε ότι χωρίζεται σε 4 μέρη-κανάλια όπου κάθε μέρος έχει τις επαφές NO, COM και NC η οποία δεν θα χρησιμοποιηθεί κάπου στο σχέδιο μας. Στη προηγούμενη υποενότητα είδαμε τι εστί φάση για κάθε συσκευή. Για να συνδεθεί κάθε συσκευή στο relay θα πρέπει πρώτα να γίνει κάποια παραμετροποίηση στο καλώδιο που συνδέονται με τη μπρίζα με πολύ προσοχή καθώς όταν θα γίνεται αυτή θα πρέπει να μην είναι συνδεδεμένες στο ρεύμα-μπρίζα. Η παραμετροποίηση για το φωτιστικό είναι η ίδια και με τις άλλες τρεις συσκευές και έχει ως εξής:

<u>ΠΡΟΣΟΧΗ</u> Ενώ η συσκευή δεν είναι συνδεδεμένη στη πρίζα.

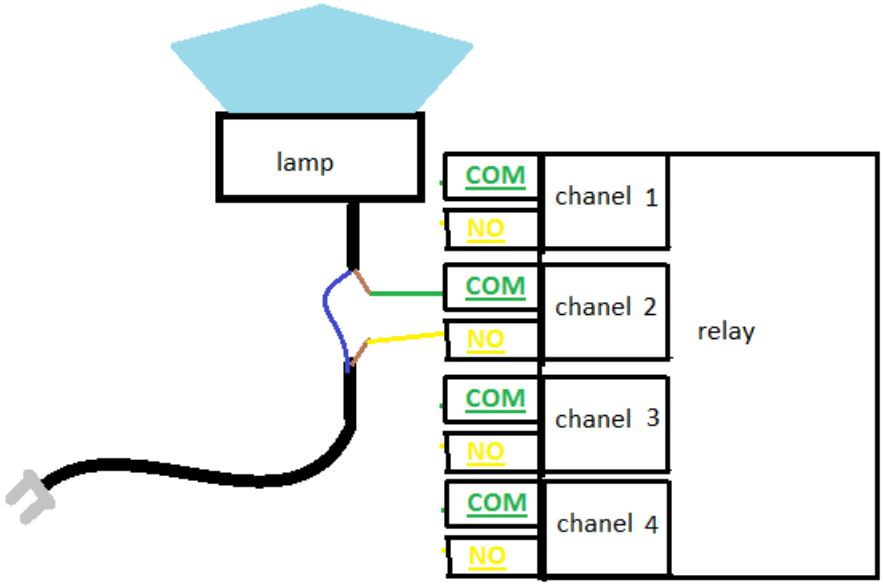
Η διαδικασία απεικονίζεται στις εικόνες που ακολουθούν. Ειδικότερα, ανοίγουμε τη βάση του φωτιστικού και κόβουμε τα δύο καλώδια, το μπλε και το καφέ. Απογυμνώνουμε τις άκρες τους και ενώνουμε μέσα στα προστατευτικά το μπλε καλώδιο του ουδετέρου. Το καφέ καλώδιο είναι της φάσης (*εικόνα* 23).

.

(Εικόνα 23. Παραμετροποίηση Φωτιστικού)

Αφού γίνει η παραμετροποίηση τους στη θύρα No του relay θα πάει η κομμένη άκρη του καφέ καλωδίου της φάσης που είναι πιο κοντά στη μπρίζα ενώ στο COM η άλλη

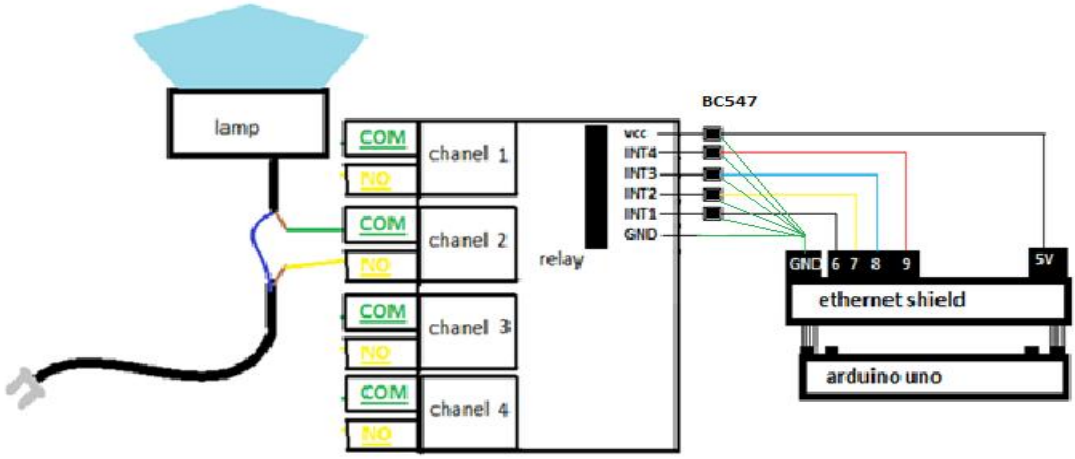
κομμένη άκρη. Επειδή οι συσκευές μας δεν μπορούν όλες να είναι τόσο κοντά στο relay και θα υπάρξει πρόβλημα στην πράξη, θα χρησιμοποιήσουμε μονωτική ταινία και καλώδιο PVC 3X1,5mm2. Την κάθε άκρη του καφέ καλωδίου ξεγυμνόνωντας την άκρη της και ενώ πάντα προσέχουμε η συσκευή μας να μην είναι συνδεδεμένη στη μπρίζα ή άλλη πηγή ρεύματος, την ενώνουμε με ένα κομμάτι καλωδίου PVC 3X1,5mm2 όσου μήκους χρειάζεται για να φτάνει στη θύρα No και COM (*Εικόνα* 24)



(Εικόνα 24, Τρόπος Σύνδεσης Φάσης με chanel)

4.6 ΤΕΛΙΚΟ ΣΧΕΔΙΟ

Το τελικό σχέδιο θα είναι αυτό που θα ακολουθήσουμε και βάση αυτών των συνδέσεων θα γίνει ο προγραμματισμός. Παρουσιάζεται στο παρακάτω σχήμα (*Εικόνα* 25)για τη συσκευή της λάμπας και τα ίδια ισχύουν και για τις άλλες τρεις.



(Εικόνα25. Τελικό σχέδιο)

Τα στοιχεία που θα χρειαστούμε είναι τα εξής:

- ARDUINO UNO
- ETHERNET SHIELD
- ΟΙ ΤΕΣΣΕΡΙΣ ΣΥΣΚΕΥΕΣ ΜΑΣ
- RELAY MODULE 4 CHANNEL
- 5 Transistor BC547
- ΚΑΛΩΔΙΑ JUMPER
- ΜΟΝΩΤΙΚΗ ΤΑΙΝΙΑ
- ΚΑΛΩΔΙΟ PVC 3X1,5mm2
- ΚΑΛΩΔΙΟ ETHERNET
- ΚΑΛΩΔΙΟ ΦΟΡΤΙΣΤΗ ΚΑΙ USB
- ΚΙ ΑΠΟ ΕΡΓΑΛΕΙΑ ΕΝΑ ΜΑΧΑΙΡΑΚΙ ΝΑ ΚΟΒΕΙ, ΕΝΑ ΜΙΚΡΟ ΚΑΤΣΑΒΙΔΑΚΙ ΣΤΑΥΡΩΤΟ.

Αφού έχουμε στη διάθεση μας όλα τα άνω στοιχεία συνδέουμε το Arduino UNO με τη SHIELD, τους ακδροδέκτες της shield GND, PINS 6,7, 8,9 και 5V με τα GND, INT 1,2,3,4 και VCC αντίστοιχα του relay αφού παρεμβάλουμε τα τρανζίστορ BC547 ενώ αφού κάνω τη παραμετροποίηση των συσκευών χωρίς αυτές να είναι συνδεδεμένες στο ρεύμα, συνδέω τις κομμένες άκρες της φάσης-καφέ καλωδίου και αυτές με τη σειρά τους στο NO ΚΑΙ COM του καναλιού πάνω στο relay (No με άκρη πιο κοντά στη μπρίζα και η άλλη με COM) . Ύστερα συνδέουμε την πλατφόρμα με τον υπολογιστή και γράφουμε το πρόγραμμα μας στο περιβάλλον ARDUINO IDE, αφού το γράψουμε το κάνουμε compile και upload, ξεσυνδέουμε τη πλατφόρμα UNO από τον υπολογιστή, συνδέουμε την ethernet shield με καλώδιο router με το modem και την Arduino UNO με εξωτερική τροφοδοσία (φορτιστή τηλεφώνου USB) και όχι με τον υπολογιστή. Μένει να αναφερθούμε στο κώδικα που θα χρησιμοποιηθεί και θα ανεβαστεί μέσω του IDE και τον πρόγραμμα για την εφαρμογή android.

ΚΕΦΑΛΑΙΟ 5 ΣΧΕΔΙΑΣΗ ΕΦΑΡΜΟΦΗΣ - ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

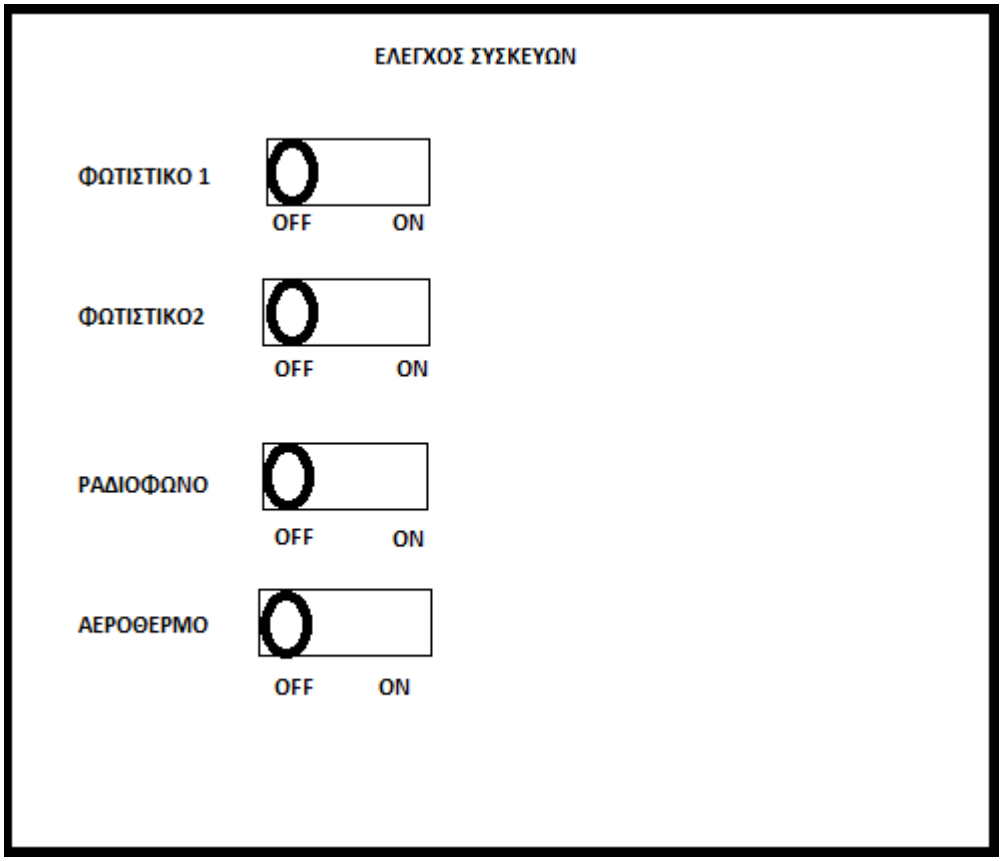
5.1 ΑΝΑΚΕΦΑΛΑΙΩΣΗ-ΕΙΣΑΓΩΓΗ

Ανακεφαλαιώνοντας έχουμε μιλήσει για το υλικό και τα συστατικά στοιχεία που θα χρησιμοποιήσουμε και έχουμε εισχωρήσει αρκετά σε πληροφορίες για το κάθε ένα. Ξέρουμε το πώς θα τα συνδέσουμε. Επίσης έχει παρουσιαστεί το λογισμικό-εργαλεία που θα χρειαστούμε για να γράψουμε τα προγράμματα μας.

Σε αυτό το κεφάλαιο, θα σχεδιάσουμε την εφαρμογή μας παρουσιάζοντας το επιθυμητό αποτέλεσμα που θέλουμε να έχουμε στο τέλος. Υστερα, θα κάνω εισαγωγή στις γλώσσες προγραμματισμού που θα χρησιμοποιηθούν στο περιβάλλον του ARDUINO και του ANDROID παρουσιάζοντας τη σύνταξη τους τις εντολές τους και τέλος ο τελικός κώδικας μαζί με τη σύνθεση και τελικό αποτέλεσμα του που θα χρησιμοποιηθεί..

5.2 ΓΕΝΙΚΗ ΑΠΟΨΗ ΣΧΕΔΙΟΥ ΕΦΑΡΜΟΓΗΣ

Πριν ξεκινήσω να γράφω το κώδικα φαντάστηκα το πώς περίπου θα ήθελα να είναι το περιβάλλον και η λειτουργία της εφαρμογής μου. Θεωρώ ότι είναι σημαντικό σε αυτό το σημείο να αναφέρω ότι η μέχρι τώρα ανάλυση του σχεδίου μου και οι πληροφορίες που παραθέτω κατά την άποψη μου δεν είναι επαρκείς ώστε να παρουσιάσω κάποια πολύπλοκη εφαρμογή καθώς νομίζω ότι στο τέλος δεν θα καταφέρω να επεξηγείται στο σωστό βαθμό πως έγινε ή πως θα γίνει. Ψάχνοντας πληροφορίες στο διαδίκτυο, άλλες πτυχιακές και παρόμοια σχέδια, παρατήρησα ότι πολλές φορές παρουσιαζόταν κάποιο πολύπλοκο σχέδιο με έναν πολύ γενικό τρόπο. Αυτό ήταν και το έναυσμα να αντιληφθώ ότι προτιμώ να υλοποιήσω κάτι απλό και κατανοητό ώστε να αποτελεί κομμάτι η σκαλοπάτι κάποιου που θέλει να υλοποιήσει κάτι πιο πολύπλοκο παρά κάτι που δεν θα εξηγείται με το σωστό τρόπο. Βασισμένος σε αυτή την αντίληψη λοιπόν σχεδίασα την εφαρμογή μου (Εικόνα 26).



(Εικόνα 26. Επιθυμητό αποτέλεσμα)

Άσχετα από τα χρώματα που θα επιλεχθούν, η οθόνη της έξυπνης συσκευής κατά το αρχικευκό σχέδιο εμφανίζει μια επικεφαλίδα «ΕΛΕΓΧΟ ΣΥΣΚΕΥΩΝ» και ύστερα αποτελείται από τέτοια κουμπιά με την εξήγηση τους αριστερά.
Όταν ο χρήστης θα σέρνει με το δάχτυλο τη μπάρα από OFF στο ON η συσκευή που αναγράφεται στην εξήγηση δίπλα θα ανάβει και το αντίθετο με την αντίθετη κίνηση της μπάρας από το ON στο OFF. Το αντίστοιχο pin του arduino στέλνει ή σταματάει να στέλνει 5V στο επίσης αντίστοιχο module-chanel που είναι συνδεδεμένη η αντίστοιχη συσκευή.

5.3 ΓΛΩΣΣΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

5.3.1 ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ARDUINO

Όπως έχουμε αναφέρει τα προγράμματα για τις λειτουργίες του arduino γράφονται κατά τη σύνδεση της πλατφόρμας σε έναν υπολογιστή που είναι εγκατεστημένο το απαραίτητο λειτουργικό IDE. Τα προγράμματα γράφονται στο περιβάλλον IDE Τα Arduino προγράμματα είναι γραμμένα σε C ή C++. Το Arduino IDE έρχεται με μια βιβλιοθήκη λογισμικού που ονομάζεται "Wiring" από το πρωτότυπο σχέδιο Wiring γεγονός που καθιστά πολλές κοινές λειτουργίες εισόδου/εξόδου πολύ πιο εύκολες.
Οι συναρτήσεις που χρησιμοποιεί το περιβάλλον ανάπτυξης είναι παρόμοιες με τις συναρτήσεις της γλώσσας C και εμπλουτισμένες με ειδικές συναρτήσεις για την διαχείριση του arduino. Μερικές συναρτήσεις παρουσιάζονται στον πίνακα 3που ακολουθεί.

LOW	Σταθερά	Int	-	Έχει την τιμή 0 και είναι αντίστοιχη του λογικού false.
HIGH	Σταθερά	Int	-	Έχει την τιμή 1 και είναι αντίστοιχη του λογικού true.
INPUT	Σταθερά	Int	-	Έχει την τιμή 0 και είναι αντίστοιχη του λογικού false.
OUTPUT	Σταθερά	Int	-	Έχει την τιμή 1 και είναι αντίστοιχη του λογικού true.
pinMode	Εντολή	-	(*pin*, *mode*)	Καθορίζει αν το συγκεκριμένο ψηφιακό *pin* θα είναι pin εισόδου ή pin εξόδου ανάλογα με την τιμή που δίνεται στην παράμετρο *mode* (INPUT ή OUTPUT αντίστοιχα).
digitalWrite	Εντολή	-	(*pin*, *pinstatus*)	Θέτει την κατάσταση *pinstatus* (HIGH ή LOW) στο συγκεκριμένο ψηφιακό *pin*.
digitalRead	Συνάρτηση	Int	(*pin*)	Επιστρέφει την κατάσταση του συγκεκριμένου ψηφιακού *pin* (0 για LOW και 1 για HIGH) εφόσον αυτό είναι pin εισόδου.
analogReference	Εντολή	-	(*type*)	Δέχεται τις τιμές DEFAULT, INTERNAL ή EXTERNAL στην παράμετρο *type* για να καθορίσει την τάση αναφοράς (V_{ref}) των αναλογικών εισόδων (5V, 1.1V ή η εξωτερική τάση
analogRead	Συνάρτηση	int	(*pin*)	Επιστρέφει έναν ακέραιο από 0 έως 1023, ανάλογα με την τάση που τροφοδοτείται το συγκεκριμένο *pin* αναλογικής εισόδου στην κλίμακα 0 ως V_{ref}.

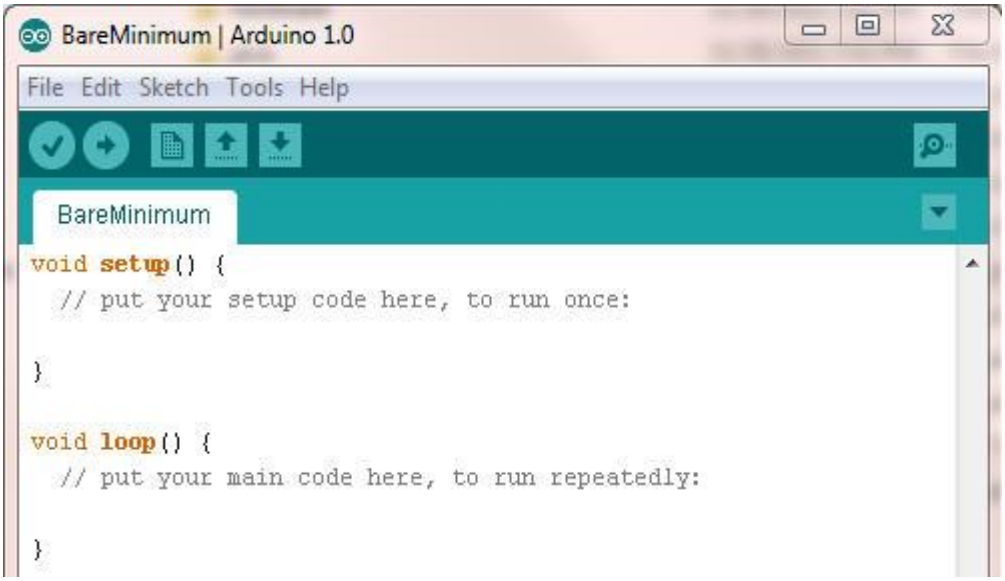
analogWrite	Εντολή	-	(*pin*, *value*)	Θέτει το συγκεκριμένο ψηφιακό *pin* σε κατάσταση ψευδο αναλογικής εξόδου (PWM). Η παράμετρος *value* καθορίζει το πλάτος του παλμού σε σχέση με την περίοδο του παραγόμενου σήματος στην κλίμακα από 0 ως 255 (π.χ. με *value* 127, το πλάτος του παλμού είναι ίσο με μισή περίοδο).
millis	Συνάρτηση	unsigned long	()	Μετρητής που επιστρέφει το χρονικό διάστημα σε ms από την στιγμή που άρχισε η εκτέλεση του προγράμματος. Λάβετε υπόψη ότι λόγω του τύπου μεταβλητής (unsigned long δηλ. 32bit) θα γίνει overflow σε 2^32ms δηλαδή περίπου σε 50 μέρες, οπότε ο μετρητής θα ξεκινήσει πάλι από το μηδέν.
delay	Εντολή	-	(*time*)	Σταματά προσωρινά την ροή του προγράμματος για *time* ms. Η παράμετρος *time* είναι unsigned long (από 0 ως 2^32). Σημειώστε ότι παρά την προσωρινή παύση, συναρτήσεις των οποίων η εκτέλεση ενεργοποιείται από interrupt θα εκτελεστούν κανονικά κατά την διάρκεια μιας καθυστέρησης (delay).

attachInterrupt	Εντολή	-	(*interrupt*, *function*, *triggermode*)	Θέτει σε λειτουργία το συγκεκριμένο *interrupt*, ώστε να ενεργοποιεί την συνάρτηση (*function)*, κάθε φορά που ικανοποιείται η συνθήκη που ορίζεται από την παράμετρο *triggermode*: • LOW (ενεργοποίηση όταν η κατάσταση του pin που αντιστοιχεί στο συγκεκριμένο interrupt γίνει
noInterrupts	Εντολή	-	()	Σταματά προσωρινά την λειτουργία όλων των interrupt
interrupts	Εντολή	-	()	Επαναφέρει την λειτουργία των interrupt που διακόπηκε προσωρινά από μια εντολή noInterrupts.
Serial.begin	Μέθοδος κλάσης	-	(*datarate*)	Θέτει τον ρυθμό μεταφοράς δεδομένων του σειριακού interface (σε baud)
Serial.println	Μέθοδος κλάσης	-	(*data*)	Διοχετεύει τα δεδομένα *data* για αποστολή μέσω του σειριακού interface. Η παράμετρος *data* μπορεί να είναι είτε αριθμός είτε αλφαριθμητικό.

(*Πίνακας 3. Εντολές ARDUINO)*

Η δομή κάθε προγράμματος του arduino περιέχει δύο συναρτήσεις την setup και την loop. Ειδικότερα, η συνάρτηση setup() εκτελείται στην αρχή του προγράμματος και για μία μόνο φορά. Περιλαμβάνει τις αρχικοποιήσεις των μεταβλητών, τις δηλώσεις των ακροδεκτών π.χ. αν ένας ακροδέκτης είναι εισόδου ή εξόδου και τις αρχικοποιήσεις των βιβλιοθηκών.

Ενώ η συνάρτηση loop(), είναι ένας βρόγχος επανάληψης, οπότε ο κώδικας που γράφεται μέσα στη συνάρτηση αυτή επαναλαμβάνεται συνεχώς δίνοντας την δυνατότητα στο πρόγραμμα μας να αλλάζει τιμές και το Arduino να ανταποκρίνεται ανάλογα .



Παράδειγμα 1

Ας δούμε ένα παράδειγμα που περιέχει τη σύνταξη ενός προγράμματος που αφορά τον καθορισμό ενός pin ως εξόδου. Συγκεκριμένα θέλουμε το pin 13 να τροφοδοτήσει με 5 V ένα Led και ύστερα από 1000 millisecond να σβήσει. Αυτό επαναλαμβάνεται συνεχώς καθώς δεν υπάρχει κάποια άλλη συνθήκη στο βρόγχο της επανάληψης « void loop ». Σημειώνεται ότι στις περισσότερες Arduino Boards υπάρχει ενσωματωμένο led πάνω σε αυτές δίπλα από το pin 13 που λειτουργεί σαν ένα led που είναι συνδεδεμένο στο pin 13. Αυτό βοηθάει το παράδειγμα μας ώστε να μην χρειάζεται να κάνουμε περαιτέρω παραμετροποιήσεις όπως να συνδέσουμε με καλώδιο ένα led στο pin 13 (υπάρχει ήδη).
Το πρόγραμμα είναι το εξής:

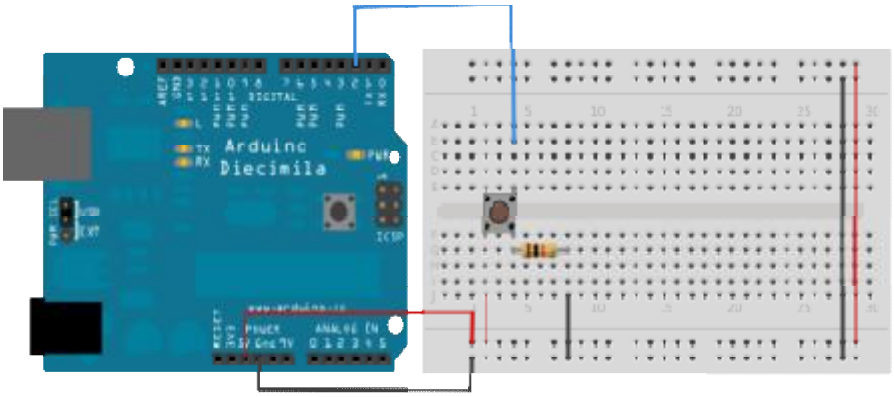
```
int ledPin = 13;                    // LED συνδεδεμένο στο digital pin 13

void setup()
{
 pinMode(ledPin, OUTPUT);           // θέτει το digital pin σαν έξοδο
}

void loop()
{
 digitalWrite(ledPin, HIGH);  // θέτει το LED on
 delay(1000);                       // περιμένουμε για ένα(1) sec
 digitalWrite(ledPin, LOW);         // θέτει το LED off
 delay(1000);          // περιμένουμε για ένα(1) second και επαναλαμβάνεται το πρόγραμμα
}
```

Το HIGH και το LOW ορίζει την ύπαρξη τροφοδότησης από το pin 13 και την μη ύπαρξη αντίστοιχα στο led ώστε να ανάβει και να σβήνει μετά από ένα δευτερόλεπτο.

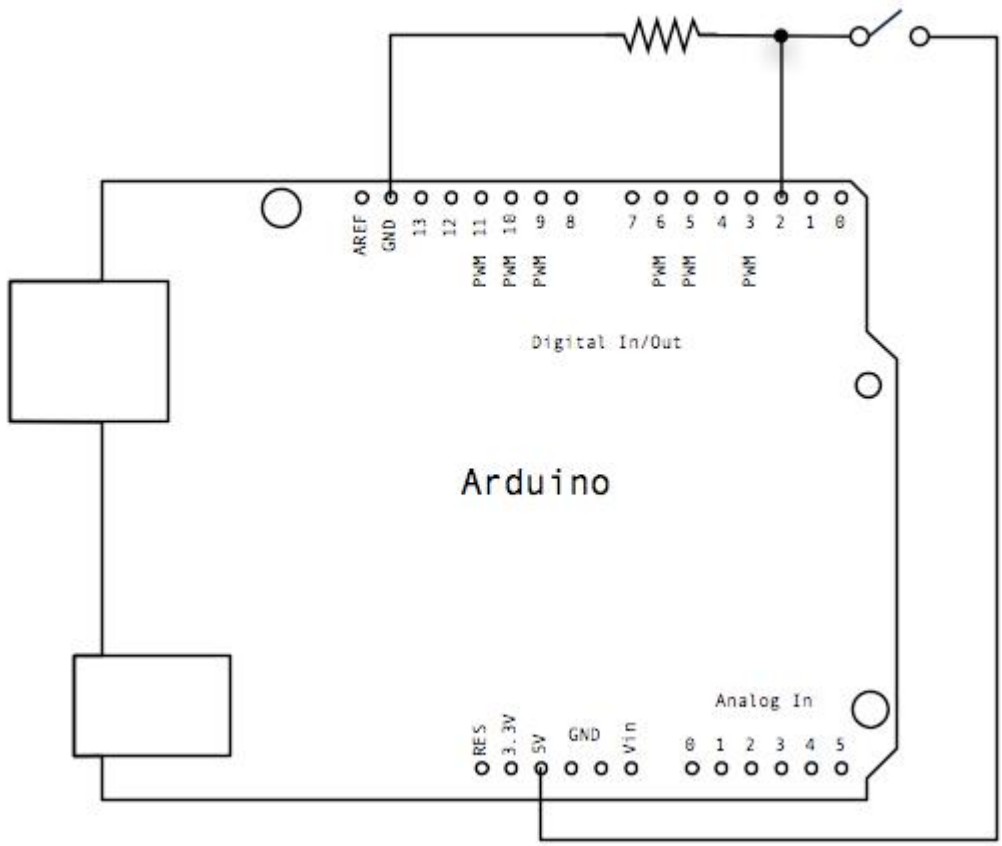
Παράδειγμα 2

Στο δεύτερο παράδειγμα μας θα χρησιμοποιήσουμε την συνθήκη if. Στο παρακάτω κύκλωμα το ψηφιακό pin 2 διαβάζει την ύπαρξη ρεύματος η οποία εξαρτάται από το αν το pushbutton είναι πατημένο ή όχι (όταν είναι πατημένο υπάρχει ρεύμα ενώ όταν δεν είναι δεν υπάρχει. Στη σύνταξη του προγράμματος η μεταβλητή pushstation ορίζεται σε HIGH ή LOW ανάλογα με το τι θα διαβάσει το ψηφιακό pin 2(ύπαρξη

ρεύματος και μη). Η συνθήκη If ορίζει ότι αν η pushstation είναι High το ψηφιακό pin 13 θα στείλει ρεύμα και θα ανάψει το led. Όπως αναφέραμε και στο παράδειγμα 1 υπάρχει ήδη ένα led ενσωματωμένο πάνω στην πλατφόρμα. Πριν δούμε το πρόγραμμα ας δούμε το κύκλωμα.



Συνδέουμε τα τρία καλώδια στον πίνακα (λευκή πλατφόρμα). Τα δύο πρώτα, κόκκινο(ξεκινάει από 5 V και τροφοδοτεί με 5v το κύκλωμα) και μαύρο (GND: γείωση εκεί που καταλήγει το ρεύμα), συνδέονται με τις δύο μεγάλες κάθετες σειρές από την πλευρά του breadboard(λευκή πλατφόρμα: διευκολύνει την σύνδεση των αγωγών ώστε να μην υπάρχουν διακοπές στο κύκλωμα καθώς μένουν σταθεροί και ακίνητοι) οι οποίες συνδέονται μεταξύ τους αφού μεσολαβεί το pushbutton. Οι συνδέσεις των δύο καλωδίων (κόκκινο και μαύρο) πάνω στο pushbutton δεν φαίνονται στο παραπάνω κύκλωμα καθώς γίνονται στα δύο «πόδια» του pushbutton τα οποία προεξέχουν στην κάτω αντίθετη επιφάνεια της breadboard από αυτή που φαίνεται στο παραπάνω κύκλωμα. Η ψηφιακή καρφίτσα 2(pin 2) συνδέεται μέσω μιας αντίστασης pull-down (εδώ 10K ohm) με το pushbutton στο ίδιο πόδι που συνδέεται και το μαύρο καλώδιο.
Όταν το κουμπί στο pushbutton είναι ανοιχτό (μη πιεσμένο) δεν υπάρχει σύνδεση μεταξύ των δύο ποδιών του μπουτόν με αποτέλεσμα να μην περνάει το ρεύμα (μέσα από την αντίσταση pull-down) και το pin 2 να διαβάζει ένα LOW. Όταν το κουμπί είναι κλειστό (πατημένο), κάνει μια σύνδεση μεταξύ των δύο ποδιών του, συνδέοντας το pin για 5 βολτ με αποτέλεσμα το pin 2 να διαβάζει ένα HIGH.
Το παρακάτω σχήμα εξηγεί καλύτερα τη συνδεσμιολογία.



ΚΩΔΙΚΑΣ

Ο Κώδικας για το παράδειγμα μας είναι ο εξής:

```
const int buttonPin = 2;     // ο αριθμός του pin που λειτουργεί σαν κουμπί on/off
const int ledPin =  13;      // ο αριθμός του pin που θέλουμε να στείλει 5 volt στο led

// variables will change:
int buttonState = 0;         // μεταβλητή που αποθηκεύεται είτε το HIGH είτε το LOW

void setup() {
  // καθορίζουμε το pin 13 σαν εξόδο:
  pinMode(ledPin, OUTPUT);
  // καθορίζουμε το pin 2 σαν είσοδο:
  pinMode(buttonPin, INPUT);
}

void loop() {
  // αποθηκεύουμε την τιμή του pin 2 στη buttonState:
  buttonState = digitalRead(buttonPin);

  // εν συνεχεία ελέγχουμε αν είναι πατημένο το κουμπί
  // Αν είναι πατημένο, η μεταβλητή  buttonState είναι HIGH:
  if (buttonState == HIGH) {
    // ανάβει το led:
    digitalWrite(ledPin, HIGH);
  } else {
    //αλλιώς το led σβήνει:
    digitalWrite(ledPin, LOW);
```

```
 }
}
```

5.3.2 HTM

Για το γραφικά στοιχεία της εφαρμογής θα χρησιμοποιηθεί και ο κώδικας HTML. Αρχικά, Με την HTML μπορούν να υλοποιηθούν τα κουμπιά και η δόμηση της εφαρμογής. Μπορεί να χρησιμοποιηθεί και στο κώδικα της εφαρμογής αλλά και στο κώδικα του ARDUINO. Το πολύ θετικό για αυτή τη γλώσσα προγραμματισμού βρίσκεται στη πολλή απλή σύνθεση καθώς και στο ότι μπορεί να χρησιμοποιηθεί σε άλλες γλώσσες προγραμματισμού. Η HTML κατά τη σύνταξη της μπορεί και αυτεί να καλεί βιβλιοθήκες όπως η jQuery, η οποία είναι μια βιβλιοθήκη JavaScript που δημιουργήθηκε το 2006. Είναι σχεδιασμένη να απλοποιεί την υλοποίηση σεναρίων (scripting) στη πλευρά του πελάτη (client-side) της HTML.
Κάποιες γενικές πληροφορίες για την HTML παρατίθονται στην επόμενη παράγραφο.

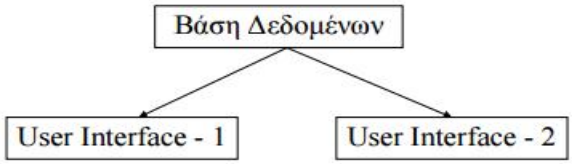
Η HTML (HyperText Markup Language ή Γλώσσα Σήμανσης Υπερκειμένου) αποτελεί μια γλώσσα σήμανσης ιστοσελίδων, η οποία δίνει τα δομικά στοιχεία των ιστοσελίδων. Βασικά στοιχεία της σύνταξής της αποτελούν οι ετικέτες που χρησιμοποιούνται ανά ζεύγη και δίνουν τα χαρακτηριστικά τους στα στοιχεία που περιέχουν. Οι ετικέτες αυτές μπορούν να ορίζουν μια ενέργεια, μια μορφοποίηση ένα είδος αντικειμένου κ.α. Με την χρήση των ετικετών τοποθετούνται τα αντικείμενα της ιστοσελίδας, όπως το κείμενο, η εικόνα, οι φωτογραφίες και δίνονται οι διαστάσεις, οι μορφοποιήσεις κ.α. Ο φυλλομετρητής (web browser), διαβάζει τα έγγραφα HTML και συνθέτει τις ιστοσελίδες. Για την συγγραφή σε γλώσσα html μπορεί να χρησιμοποιηθεί ένα απλό αρχείο κειμένου ή να χρησιμοποιηθούν ειδικά λογισμικά που προσφέρουν πρόσθετες ιδιότητες και ευκολίες, όπως το dreamweaver. Οι ετικέτες δεν εμφανίζονται, αλλά χρησιμοποιούνται από τον φυλλομετρητή για να ερμηνεύσει το περιεχόμενο της ιστοσελίδας. Συνήθως, η γλώσσα HTML χρησιμοποιείται με γλώσσες που προσφέρουν ενέργειες και αλληλεπίδραση μέσω σεναρίων, όπως η JavaScript. Ειδικότερα, μια γλώσσα scripting είναι μια ελαφριά γλώσσα προγραμματισμού που υποστηρίζει τη συγγραφή σεναρίων. Σενάρια είναι γραμμές κώδικα που μπορούν να ερμηνεύονται και να εκτελούνται χωρίς μεταγλώττιση. Στην HTML εισάγεται σύνδεση ή απλά ενσωματώνονται τα CSS (Cascading Style Sheets, Διαδοχικά Φύλλα Στυλ). Τα CSS χρησιμοποιούνται για τον έλεγχο της εμφάνισης ενός εγγράφου. Τα CSS είναι μια γλώσσα που δίνει την δυνατότητα διαμόρφωσης των χρωμάτων της στοίχισης κ.α., είτε για ξεχωριστά χαρακτηριστικά της σελίδας, είτε για την ίδια την σελίδα. Η PHP είναι μια γλώσσα προγραμματισμού, βασισμένη σε σενάρια. Χρησιμοποιείται για την ανάπτυξη διαδικτυακών εφαρμογών και προσφέρει ένα πλήθος συναρτήσεων. Ακόμα, δίνει τη δυνατότητα για σύνδεση και διαχείριση των βάσεων δεδομένων. Η Javascript είναι αντίστοιχη της γλώσσας php. Χρησιμοποιείται σε ιστοσελίδες, με την συνεργασία της html. Η javascript δημιουργεί και εκτελεί σενάρια και η HTML τα παρουσιάζει. Βασικές λειτουργίες της javascript είναι ο χειρισμός CSS, η δυναμική αλλαγή ετικετών και περιεχομένου HTML, η αποθήκευση και η ανάκτηση πληροφοριών στον

υπολογιστή του χρήστη, η εκτέλεση μετά από συμβάν και η επικοινωνία με php, xml, json αρχεία, αλλά και με άλλες ιστοσελίδες.

5.3.3 JAVA

5.3.3.1 ΛΙΓΙΑ ΛΟΓΙΑ ΓΙΑ ΤΟΝ ΑΝΤΙΚΕΙΜΕΝΟΣΤΡΑΦΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟ

Η φιλοσοφία που ακολουθεί κάποιος για να φτιάξει μια εφαρμογή βασίζεται στον αντικειμενοστραφή προγραμματισμό. Ο αντικειμενοστραφής προγραμματισμός, (OOP - Object Oriented Programming), είναι μια προγραμματιστική φιλοσοφία όπως και ο προστακτικός ή ο λογικός προγραμματισμός. Σύμφωνα με τον OOP ένα πρόγραμμα ΔΕΝ αποτελείται από τα δεδομένα και τον κώδικα που τα επεξεργάζεται αλλά από αντικείμενα (objects) τα οποία εμπεριέχουν τα δεδομένα και τα οποία (αντικείμενα) ανταλλάσσουν μεταξύ τους πληροφορίες και μηνύματα προκειμένου να επιτευχθεί ο στόχος του προγράμματος. Για παράδειγμα έστω ότι έχουμε ένα σύστημα που περιλαμβάνει μια βάση δεδομένων καθώς και user interfaces με τα οποία οι χρήστες επικοινωνούν και αλληλεπιδρούν με την Β.Δ..



Τα UIs στέλνουν τις εντολές των χρηστών στην ΒΔ και παρουσιάζουν στην οθόνη τις απαντήσεις που λαμβάνουν. Δηλαδή όλο το πρόγραμμα αποτελείται από ένα αντικείμενο ΒΔ και 2 αντικείμενα UI τα οποία είναι ολόιδια (αλλά εξυπηρετούν άλλους χρήστες). Συνεπώς ο κώδικας που θα γράφαμε θα περιείχε τον ορισμό ενός αντικειμένου ΒΔ, τον ορισμό ενός αντικειμένου UI και την κατασκευή ενός ΒΔ και 2 UI αντικειμένων.

Ορολογία

Ο κώδικας που ορίζει ένα αντικείμενο λέγεται κλάση (class) του αντικειμένου αυτού. Η κλάση χρησιμοποιείται σαν μήτρα για την κατασκευή πανομοιότυπων αντιγράφων - αντικειμένων. Τα αντικείμενα - αντίγραφα, λέγονται στιγμιότυπα (instances) της κλάσης αυτής και η κατασκευή και αρχικοποίηση ενός από αυτά λέγεται instantiation.

Σημείωση

Το αντικείμενο αυτό καθ' εαυτό είναι μια οντότητα στη μνήμη η οποία περιέχει δεδομένα καθώς και μεθόδους μέσω των οποίων μπορούμε να αλλάξουμε τα δεδομένα ή να επικοινωνήσουμε με το αντικείμενο. Αντίθετα η κλάση είναι απλώς ένα πρότυπο για την δημιουργία αντιγράφων του ίδιου αντικειμένου. Είναι δηλαδή ότι είναι ο Τύπος Δεδομένων για τις μεταβλητές (στον δομημένο προγραμματισμό).

5.3.3.2 JAVA

Όπως ειπώθηκε και παραπάνω τα αντικείμενα περιέχουν δεδομένα και παρέχουν μεθόδους για την επεξεργασία των δεδομένων αυτών καθώς και για την επικοινωνία με άλλα αντικείμενα. Ας εμβαθύνουμε λίγο περισσότερο σε αυτές τις έννοιες. Κατ` αρχήν τα δεδομένα μπορεί να είναι άλλα αντικείμενα που περιέχονται μέσα σε ένα άλλο αντικείμενο ή μπορεί να είναι κοινές μεταβλητές όπως τις γνωρίζουμε από την C και την PASCAL. Τις μεταβλητές και τα αντικείμενα αυτά τα καλούμε πεδία ή instance variables του αντικειμένου. Σε αυτό το σημείο να παρατηρήσουμε ότι μία κλάση μοιάζει με ένα structure της C το οποίο μπορεί να περιέχει κοινές μεταβλητές ή μεταβλητές που προέκυψαν από άλλα structures. Επιπλέον μία κλάση (ή ένα αντικείμενο) περιέχει και μεθόδους για την

επικοινωνία του με τον έξω κόσμο, δηλαδή τα άλλα αντικείμενα. Οι μέθοδοι αυτοί υλοποιούνται σαν συναρτήσεις παρόμοιες με αυτές της C. Όταν λοιπόν κάποιος θέλει να ζητήσει κάτι από ένα αντικείμενο, (ή εναλλακτικά να στείλει ένα μήνυμα - αίτημα), δεν έχει παρά να καλέσει - εκτελέσει την αντίστοιχη μέθοδο του αντικειμένου. Αυτό γίνεται πολύ απλά ως εξής : .(); Π.χ. mycar.startEngine(); Αυτό το σχήμα μας θυμίζει πολύ τον τρόπο πρόσβασης σε μία μεταβλητή που περιέχεται σε ένα structure της C. Επίσης να σημειώσουμε ότι μια μέθοδος μπορεί να καλεί άλλες μεθόδους του ίδιου αντικειμένου ή να επεξεργάζεται τα πεδία του. Φυσικά μπορεί να καλεί και μεθόδους ξένων αντικειμένων εφόσον έχει κάποιον δείκτη σε αυτά. ΣΗΜΕΙΩΣΗ Εδώ είδαμε ένα πολύ σπουδαίο χαρακτηριστικό του OOP. Αυτό είναι το να μπορεί να κρατά τα δεδομένα του κρυφά από τα άλλα αντικείμενα αλλά και να προσφέρει μεθόδους με τις οποίες μπορούν άλλα αντικείμενα να επικοινωνούν και να αλληλεπιδρούν μαζί του. Επιπλέον ο κώδικας που υλοποιεί αυτές τις μεθόδους είναι άγνωστος σε άλλες κλάσεις ή αντικείμενα. Έτσι έχουμε την δημιουργία ενός interface επικοινωνίας ανεξάρτητου από την υλοποίηση και την εσωτερική δομή του αντικειμένου. Αυτό λέγεται implementation hiding που είναι μία μορφή data abstraction.

Κάποια από τα στοιχεία της JAVA παρουσιάζονται παρακάτω.

Κληρονομικότητα (inheritance)

Είναι ο μηχανισμός εκείνος ο οποίος επιτρέπει σε μια κλάση B να κληρονομήσει πεδία και μεθόδους από μια κλάση A. Λέμε ότι η "B κληρονομεί από την A". Τα αντικείμενα (instances) της κλάσης B έχουν πρόσβαση στα δεδομένα και τις μεθόδους της κλάσης A χωρίς να χρειάζεται να τα ξαναδηλώσουμε. Δηλαδή είναι σαν να αντιγράψαμε τα περιεχόμενα της κλάσης A στην κλάση B. Ακόμη, να πούμε ότι για να δηλώσουμε ότι η B κληρονομεί από την A χρησιμοποιούμε την δεσμευμένη λέξη extends : Ένα παράδειγμα σε java :

```
class A {
 int a;
void add(int x) {
 a += x;} }
class B extends A {
// Τα 'a' και 'void add(int x) {...}' υπονοούνται.
int prev;
void sub(int x) {
prev = a; a -= x; }
} B
```

Υπερφόρτωση (overloading)

Πριν περάσουμε στους τρόπους δημιουργίας αντικειμένων ας δούμε το παρακάτω κομμάτι κώδικα, το οποίο είναι σωστό.

```
class number {
 int a;
 void add(int x) {
 a += x;
}
 void add(float f) {
 a += ((int) f);
}
void add(number n) {
 a += n.get_a(); }
int get_a() {
```

```
return a;
} }
```
Βλέπουμε δηλαδή ότι η συνάρτηση void add(...) χρησιμοποιείται πολλές φορές (3) αλλά κάθε φορά με διαφορετική λίστα παραμέτρων. Αυτή η επαναχρησιμοποίηση του ίδιου ονόματος λέγεται υπερφόρτωση (overloading) του ονόματος αυτού. Στις object oriented γλώσσες επιτρέπεται ο ορισμός μίας μεθόδου με τι ίδιο όνομα δύο ή περισσότερες φορές αρκεί να έχουν διαφορετικές λίστες παραμέτρων. Παράδειγμα : aMethod(), aMethod(type1), aMethod(type2), aMethod(type1, type2), aMethod(type2, type1). Όλες οι παραπάνω μέθοδοι είναι διαφορετικές μεταξύ τους. Έτσι, κατά την κλήση μιας μεθόδου κατ' αρχήν εξετάζεται το όνομά της και έπειτα εξετάζεται και η λίστα των παραμέτρων της, όσον αφορά το πλήθος, τον τύπο και τη σειρά των παραμέτρων. Αυτό είναι μια μορφή πολυμορφισμού.

Κατασκευαστές (constructors)

Κάθε κλάση διαθέτει τουλάχιστο μία μέθοδο η οποία εκτελείται κατά τη δημιουργία ενός στιγμιότυπου της, (δηλ. κατά τη δημιουργία ενός object της κλάσης αυτής). Αυτές οι μέθοδοι λέγονται κατασκευαστές (constructors) της κλάσης. Σκοπός των constructors είναι η δέσμευση μνήμης για την κατασκευή του αντικειμένου καθώς και η διενέργεια κατάλληλων αρχικοποιήσεων. Ο προγραμματιστής μπορεί να ορίσει πολλούς constructors για μία κλάση αλλά για τη δημιουργία ενός αντικειμένου (στιγμιότυπου) μπορεί να καλέσει μόνο έναν από αυτούς. Φυσικά η επιλογή είναι ελεύθερη. Αν όμως ο προγραμματιστής δεν ορίσει κανένα constructor τότε η γλώσσα καλεί έναν constructor της υπερκλάσης (κάποιον που δεν θέλει παραμέτρους) ή δίνει μήνυμα λάθους αν δεν βρει κάποιον τέτοιο. Ένα θέμα που προκύπτει είναι το πώς ορίζονται οι constructors. Στην JAVA ορίζονται όπως ακριβώς και οι απλές μέθοδοι με μόνη τη διαφορά ότι έχουν το ίδιο όνομα με την κλάση, πχ :

```
class A {
int n; A() {
// Constructor - 1 n=0;
}
A(int x) {
// Constructor - 2
n = x; }
}
```

Σε αυτήν την περίπτωση οι constructors ξεχωρίζονται μεταξύ τους από τις λίστες των παραμέτρων τους. Επίσης, οι constructors δεν έχουν επιστρεφόμενο τύπο, ούτε void. (Βλέπε το παραπάνω παράδειγμα). Η κατασκευή ενός νέου αντικειμένου γίνεται ως εξής: A a;
a = new A(); ή a = new A(5);

Προσδιοριστές πρόσβασης (access specifiers)

Τυχαίνει πολλές φορές να θέλουμε να έχουμε κάποια δεδομένα ή μεθόδους μας ορατά στα αντικείμενα όλων των κλάσεων ή να θέλουμε κάποια πεδία να είναι ορατά μόνο στα αντικείμενα της τρέχουσας κλάσης και όχι και στις υποκλάσεις της κλπ. Σε όλες αυτές τις περιπτώσεις χρησιμοποιούμε τους access specifiers. Η σύνταξη των δηλώσεων με τους access specifiers παρόντες γίνεται:

```
<access specifiers> <type or class>    <variables>;

<access specifiers> <returned type> <method name> (<param list>) {
        <body>
}
```

Οι access specifiers είναι δεσμευμένες λέξεις της JAVA οι οποίες καθορίζουν το ποιος θα έχει πρόσβαση, δηλαδή το ποιος θα μπορεί να χρησιμοποιεί και να επεξεργάζεται, τις μεθόδους ή/και τα πεδία που έχουν κάποιον τέτοιο προσδιοριστή. Κατ' αρχήν ας δούμε ποιοι είναι :

- public - Σημαίνει ότι το συγκεκριμένο πεδίο/μέθοδος μπορεί να χρησιμοποιηθεί από οποιονδήποτε, ακόμα και από ξένο αντικείμενο.
- protected - Σημαίνει ότι το πεδίο/μέθοδος που το έχει είναι ορατό μόνο στις μεθόδους της κλάσης αυτής καθώς και στις υποκλάσεις.
- private - Σημαίνει ότι το πεδίο/μέθοδος είναι ορατό μόνο στις μεθόδους αυτής της κλάσης αλλά όχι στις υποκλάσεις της.
- τίποτα - Είναι public αλλά μόνο για το τρέχων package.

Παραδείγματα :

```
 class A
 { public int a;    //φαίνεται από παντού
int b;                //φαίνεται σε όλο το package, πρακτικά παντού
 protected c;      //φαίνεται στην A και την B
private d;     //φαίνεται μόνο στην A.
 public A() {...}
public void add(int x) {....}
 void add(A a) {.....}
protected int convert_A_to_int (A a) { return a.getValue(); }
private void who_am_i() { System.out.println("I am class A!"); } }
 class B extends A {
 // I CAN SEE : a, b, c, A() via super(), add(int), add(A),
// convert_A_to_int(A), BUT I CAN'T SEE : d, who_am_i()
}
```

Επιπλέον σε κάθε πρόγραμμα θα πρέπει να υπάρχει το πολύ μία κλάση δηλωμένη ως εξής: public class A { ..... }. Αυτή η κλάση (A) θα είναι η αφετηρία του προγράμματός μας, δηλαδή αυτήν την κλάση θα πρέπει να δώσουμε από το command line στον java interpreter για να ξεκινήσει την εκτέλεση του προγράμματος. Επίσης το όνομα του αρχείου με τον πηγαίο κώδικα θα πρέπει να έχει το όνομα της κλάσης αυτής συν την επέκταση .java, (π.χ. A.java). Επιπλέον, η κλάση - αφετηρία πρέπει να περιέχει και μία μέθοδο δηλωμένη ως εξής :

```
public static void main(String args[]) {
 ..... // Το σώμα του προγράμματος γράφεται εδώ! }
```

Αυτά είναι μόνο κάποια βασικά στοιχεία της JAVA. Αποτελείται επίσης από μεταβλητές, τελεστές, βιβλιοθήκες και ανάλογα με το τι θέλουμε να επιτύχουμε κλάσεις. Με το κεφάλαιο αυτό προσπάθησα να δώσω μια πολύ γενική περιγραφφή της JAVA καθώς θεωρώ ότι η JAVA αποτελεί από μόνη της ένα πολύ μεγάλο κεφάλαιο και δεν δύναται να παρουσιαστεί σε λίγες σελίδες.

ΚΕΦΑΛΑΙΟ 6 ΠΑΡΟΥΣΙΑΣΗ

6.1 ΤΕΛΙΚΟ ΣΧΕΔΙΟ

Το τελικό σχέδιο περιλαμβάνει το πώς θα μεταφραστεί το επιζητούμενο αποτέλεσμα σε κώδικα. Στα προηγούμενα κεφάλαια έχω αναφερθεί στη συνδεσμιολογία και παραμετροποίηση όλων των συσκευών, τι χρειαζόμαστε από υλικό και λογισμικό και έχω κάνει μια μικρή αναφορά στις γλώσσες προγραμματισμού. Θεωρώ ότι μένει να συντάξουμε τους κώδικες για το IDE ARDUINO (λογισμικό arduino) και εφαρμογής.

Αυτό που θέλω να υλοποιήσω είναι να σχεδιάσω ένα περιβάλλον για την συσκευή μου με την οποία μέσω του διαδικτύου θα μεταφέρω από στον arduino uno ώστε να τι εκτελέσει. Η λογική μου είναι να εγκαταστήσω έναν HTTP server στο arduino . Επίσης, μέσα στο κώδικα του arduino χρησιμοποιώντας την html θα μεταφράσω τα επιζητούμενα γραφικά στοιχεία της εφαρμογής μου. Προγραμματιστικά υπάρχουν κι άλλοι τρόποι και ίσως ο δικός μου να μην είναι ο πιο σωστός αλλά όπως ανέφερα θα προτιμήσω αυτό που είναι πιο εύκολος για μένα και τις μέχρι τώρα γνώσεις μου.

Οσον αφορά τον κώδικα της εφαρμογής, θα ορίσω ένα web view όπου θα συνδέεται στην IP του Http server που εγκαταστήσαμε μέσω του κώδικα στον arduino.

ΚΩΔΙΚΑΣ ARDUINO UNO

ΕΠΕΝΘΥΜΙΣΗ: Όπως ανέφερα σε προηγούμενο κεφάλαιο για τη συνδεσμιολογία των συσκευών με τον ARDUINO θα χρησιμοποιήσω τα Pin 6(κόκκινο jumper),7(πράσινο jumper),8(μαύρο jumper) και 9(μπλε jumper) για να μεταφέρω 5 V όταν το επιλέγω στην εφαρμογή και να λειτουργεί αντίστοιχα η κάθε συσκευή.
Ο κώδικας για τον ARDUINO UNO είναι ο εξής:

Αρχείο «ARDUINO»

```
#include "SPI.h"        //καλώ βιβλιοθήκες που θα χρειαστώ
#include "Ethernet.h"
#include "WebServer.h"   // κάνω download τη βιβλιοθήκη webduino

static uint8_t mac[6] = { 0xAA, 0xAA, 0xAA, 0xAA, 0xAA, 0xAA };
//ορίζω διεύθυνση mac για την Ethernet shield
static uint8_t ip[4] = { 192, 168, 10, 250 }; // η ip μας

#define PREFIX "/remote"
WebServer webserver(PREFIX, 80);  //ορίζω θύρα port του δικτύου που θα ανοίξει
//αρχικοποιώ τα pin
#define RED_PIN 6
#define GREEN_PIN 7
#define BLACK_PIN 8
#define BLUE_PIN 9
```

```
int red = 0;  //pin 6, φεύγει με κόκκινο καλώδιο από το arduino πάει στο φωτιστικό 1
int green = 0;   //pin 7, φεύγει με πράσινο καλώδιο από το arduino, πάει στο φωτισ 2
int black = 0;   //pin 8, φεύγει με μαύρο καλώδιο από το arduino και πάει στο
//ραδιόφωνο
int blue = 0;//pin 9, φεύγει με μπλε καλώδιο από το arduino και πάει στο αερόθερμο
void rgbCmd(WebServer &server, WebServer::ConnectionType type, char *, bool)
{
  if (type == WebServer::POST) // HTTP request Post method
  {
    bool repeat;
    char name[16], value[16];
    do
    {
      repeat = server.readPOSTparam(name, 16, value, 16);

      if (strcmp(name, "red") == 0)//μηδέν δεν στέλνει ρεύμα το pin 6
      {
        red = strtoul(value, NULL, 10);//γραμμή τροφοδοσίας και στέλνει ρεύμα pin 6
      }
      if (strcmp(name, "green") == 0)
      {
        green = strtoul(value, NULL, 10);
      }
      if (strcmp(name, "black") == 0)
      {
        black = strtoul(value, NULL, 10);
      }
      if (strcmp(name, "blue") == 0)
      {
        blue = strtoul(value, NULL, 10);
      }
    } while (repeat);

    server.httpSeeOther(PREFIX);

    return;
  }

  server.httpSuccess();

  if (type == WebServer::GET) //HTTP request GET METHOD
  {
    P(message) =
"<!DOCTYPE html><html><head>"
  "<meta http-equiv=\"Content-Type\" content=\"text/html; charset=utf-8\">"
  "<title>ELEGXOS TESSARON SUSKEUON</title>"
  "<link  rel=\"stylesheet\"  href=\"http://code.jquery.com/mobile/1.3.1/jquery.mobile-
1.3.1.min.css\" />"
  "<script src=\"http://code.jquery.com/jquery-1.9.1.min.js\"></script>"
```

```
  "<script src=\"http://code.jquery.com/mobile/1.3.1/jquery.mobile-1.3.1.min.js\"></script>"
  "<script>"
// va
   "$(document).ready(function(){ $('#red, #green, #black, #blue').slider; $('#red, #green, #black, #blue').bind( 'change', function(event, ui) { jQuery.ajaxSetup({timeout: 1000}); /*not to DDoS the Arduino, you might have to change this to some threshold value that fits your setup*/ var id = $(this).attr('id'); var strength = $(this).val(); if (id == 'red') $.post('/remote', { red: strength } ); if (id == 'green') $.post('/remote', { green: strength } ); if (id == 'black') $.post('/remote', { black: strength } ); if (id == 'blue') $.post('/remote', { blue: strength } ); });});"
  "</script>"
"</head>"
"<body>"
  "<div data-role=\"header\" data-position=\"inline\"><h1>CONTROL DEVICE</h1></div>"
   "<div class=\"ui-body ui-body-a\">"
    "<div data-role=\"fieldcontain\">"
     "<h2>Φωτιστικό 1</h2>"
        "<label for=\"flip-3\">Διακόπτης 1</label>"
        "<select name=\"flip-3\" id=\"red\" data-role=\"slider\" data-theme=\"b\">"
                "<option value=\"0\">OFF</option>"
                "<option value=\"255\">ON</option>"
        "</select> "
     "<h2>φωτιστικό 2</h2>"
     "<label for=\"flip-3\">Διακόπτης 2</label>"
        "<select name=\"flip-3\" id=\"green\" data-role=\"slider\" data-theme=\"b\">"
                "<option value=\"0\">OFF</option>"
                "<option value=\"255\">ON</option>"
        "</select> "
     "<h2>ραδιόφωνο</h2>"
     "<label for=\"flip-3\">Διακόπτης 3</label>"
        "<select name=\"flip-3\" id=\"black\" data-role=\"slider\" data-theme=\"b\">"
                "<option value=\"0\">OFF</option>"
                "<option value=\"255\">ON</option>"
        "</select> "
     "<h2>αερόθερμο</h2>"
     "<label for=\"flip-3\">Διακόπτης 4</label>"
        "<select name=\"flip-3\" id=\"blue\" data-role=\"slider\" data-theme=\"b\">"
                "<option value=\"0\">OFF</option>"
                "<option value=\"255\">ON</option>"
        "</select> "
    "</div>"
   "<center>
<img src="http://www.greeceandroid.gr/images/Android.png"></center>"
   "</div>"
  "</body>"
"</html>";

   server.printP(message);
```

```
  }
}

void setup()
{
  pinMode(RED_PIN, OUTPUT);
  pinMode(GREEN_PIN, OUTPUT);
  pinMode(BLACK_PIN, OUTPUT);
  pinMode(BLUE_PIN, OUTPUT);
  Ethernet.begin(mac, ip);

  webserver.setDefaultCommand(&rgbCmd);

  webserver.begin();
}

void loop()
{
  webserver.processConnection();
  analogWrite(RED_PIN, red);
  analogWrite(GREEN_PIN, green);
  analogWrite(BLACK_PIN, black);
  analogWrite(BLUE_PIN, blue);
}
```

ΚΩΔΙΚΑΣ ADROID

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="demo.arduino.com.arduinodemo">
    <uses-permission android:name="android.permission.INTERNET" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
```

```
</manifest>
```

MainActivity.java

```
package demo.arduino.com.arduinodemo;

import android.app.Activity;
import android.os.Build;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.KeyEvent;
import android.view.View;
import android.view.Menu;
import android.view.MenuItem;
import android.view.Window;
import android.webkit.WebChromeClient;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;

public class MainActivity extends AppCompatActivity {

   WebView webview;

   public void onCreate(Bundle savedInstanceState) {
      this.getWindow().requestFeature(Window.FEATURE_PROGRESS);
      super.onCreate(savedInstanceState);

      getWindow().setFeatureInt(Window.FEATURE_PROGRESS,
Window.PROGRESS_VISIBILITY_ON);
      setContentView(R.layout.activity_main);


      webview = (WebView) findViewById(R.id.web_engine);
      webview.setWebViewClient(new HelloWebViewClient());
      webview.getSettings().setJavaScriptEnabled(true);

      if                    (Integer.parseInt(Build.VERSION.SDK)                    >=
Build.VERSION_CODES.FROYO)
         webview.getSettings().setPluginState(WebSettings.PluginState.ON);
      webview.getSettings().setAllowFileAccess(true);
      webview.getSettings().setBuiltInZoomControls(true);       //multitouch       an
ipostirizetai
      webview.setScrollBarStyle(WebView.SCROLLBARS_OUTSIDE_OVERLAY);
      webview.loadUrl("http://192.168.1.250/remote"); //o server tou Arduino
```

```
        //webview.loadUrl("http://google.com"); //gia debug

        final Activity MyActivity = this;
        webview.setWebChromeClient(new WebChromeClient() {
            public void onProgressChanged(WebView view, int progress)
            {
                //Na svisw tin mpara afou fortwsei to url
                MyActivity.setTitle("Φόρτωση...");
                MyActivity.setProgress(progress * 100); //svisimo

                // Kai na emfanisw to app name
                if(progress == 100)
                    MyActivity.setTitle(R.string.app_name);
            }
        });

    }

    //Na krataw ta clicks within the app
    private class HelloWebViewClient extends WebViewClient {
        @Override
        public boolean shouldOverrideUrlLoading(WebView view, String url) {
            return false;
        }
    }

    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event) {
        if ((keyCode == KeyEvent.KEYCODE_BACK) && webview.canGoBack()) {
            webview.goBack();
            return true;
        }
        return super.onKeyDown(keyCode, event);
    }
}
```

layout/activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="demo.arduino.com.arduinodemo.MainActivity">
```

```
    <WebView
        android:layout_width="wrap_content"
        android:id="@+id/web_engine"
        android:layout_height="wrap_content"
        android:text="Hello World!" />
</LinearLayout>
```